On Coding for Delay—Network Coding for Time-Division Duplexing

Daniel E. Lucani, *Member, IEEE*, Muriel Médard, *Fellow, IEEE*, and Milica Stojanovic, *Fellow, IEEE*

***Abstract*—In networks with large latency, feedback about received packets may lag considerably the transmission of the original packets, limiting the feedback's usefulness. Moreover, time duplex constraints may entail that receiving feedback may be costly. In this work, we consider tailoring feedback and coding jointly in such settings to reduce the expected delay for successful in order reception of packets. We find that, in certain applications, judicious choices provide results that are close to those that would be obtained with a full-duplex system. We study two cases of data transmission: one-to-all broadcast and all-to-all broadcast. We also analyze important practical considerations weighing the trade off between performance and complexity in applications that rely on random linear network coding. Finally, we study the problem of transmission of information under the large latency and time duplexing constraints in the presence of random packet arrivals. In particular, we analyze the problem of using a batch by batch approach and an online network coding approach with Poisson arrivals. We present numerical results to illustrate the performance under a variety of scenarios and show the benefits of the proposed schemes as compared to typical ARQ and scheduling schemes.**

***Index Terms*—Bulk queueing, half duplex, large latency, network coding, online network coding, time-division duplexing.**

I. Introduction

THIS paper constitutes a step toward coding with delay as the main focus of study and optimization. We focus on reliable communications in environments with packet erasure channels, large latency, and with nodes that have a half-duplex or time-division duplexing constraint. In particular, we focus on tailoring coding and feedback to determine how long a node should transmit before it stops to listen for other nodes' transmissions.

A. Background

The concept of network coding was introduced by Ahlswede *et al.* [1]. The fundamental idea of network coding is to encourage the system to mix different data packets at intermediate nodes through coding, rather than storing and forwarding copies of packets that are routed through the network. Under this premise, it is no longer required for the system to keep track of which packets have been received: receivers need only aim at accumulating enough coded packets in order to recover the information.

Network coding research originally studied throughput performance without delay considerations for channels with no erasures and no feedback [1]–[3]. References [2] and [3] showed linear codes over a network to be sufficient to establish any feasible multicast connection. Reference [4] proved that randomly generated linear codes in a distributed fashion also achieve multicast capacity with high probability, providing a crucial step toward a practical implementation of network coding. For networks with packet erasures, two approaches have been used. The first approach relies on rateless codes (i.e., transmission of coded packets until all terminals receive enough information to decode). Studies have shown the optimality of random linear network coding (RLNC) for multicast connections in wireline and wireless networks [5], tradeoffs between memory usage and achievable rate [6], and modifications to the code structure to preserve the communication efficiency of RLNC, while achieving better computational efficiency [7]. Practical implementations, such as MORE [8], have illustrated throughput gains of network coding via experimental results.

Delay performance gains of network coding are more recent and have focused on scenarios of large file transmissions with limited or no feedback (e.g., [9] and [10]). Other studies on delay performance have compared RLNC with Automatic Repeat reQuest (ARQ) and forward error correcting (FEC) techniques (e.g., [11]). One of the first uses of feedback in network coding was to provide efficient queue management [12]. Reference [12] used feedback to acknowledge degrees of freedom (dofs), defined as linearly independent combinations of the data packets, instead of original data packets, to show that queue size in a node follows the dofs.

The second approach focuses on block-by-block transmissions. For this approach, linear codes are shown to achieve capacity in wireless networks [13], while the use of feedback has been constrained to queueing techniques to acknowledge successful transmission of each block of data packets [14], [15].

Manuscript received April 15, 2010; revised October 15, 2010; accepted May 24, 2011. Date of current version March 13, 2012. This work was supported in part by the National Science Foundation under Grant Nos. 0520075, 0831728, and CNS-0627021; in part by ONR MURI Grant No. N00014-07-1-0738, subcontract #060786 issued by BAE Systems National Security Solutions, Inc.; and in part by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare System Center (SPAWARSYSCEN), San Diego, under Contract No. N66001-06-C-2020 (CBMANET), and in part by the European Commission under Grant FP7-INFSO-ICT-215252 (N-Crave Project). Parts of this work were presented at the 2009 Information Theory and Applications Workshop (ITA), the 2009 IEEE Conference on Computer Communications, the 2009 IEEE Symposium on Network Coding [29], the 2009 IEEE International Communications Conference [40], the 2009 IEEE International Symposium on Information Theory [36], the Annual Allerton Conference on Communication, Control, and Computing, Sept. 2009, [39], and the 2009 IEEE Global Communications Conference [38].

D. E. Lucani is with the Instituto de Telecomunicações, Departamento de Engenharia Electrotécnica e de Computadores, Faculdade de Engenharia da Universidade do Porto, Porto 4200-465, Portugal (e-mail: dlucani@fe.up.pt).

M. Médard is with the MIT Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: medard@mit.edu).

M. Stojanovic is with the ECE Department, Northeastern University, Boston, MA 02118 USA (e-mail: millitsa@ece.neu.edu).

Communicated by M. Effros, Associate Editor for the special issue on "Facets of Coding Theory: From Algorithms to Networks."

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2011.2177562

B. Motivation

We study a problem that we believe has not been considered previously: coding for delay in time-division duplexing (TDD) channels (i.e., when a node can only transmit or receive, but not both at the same time). A TDD channel is usually called half duplex, but we prefer the term TDD to emphasize that the channel is not assigned half of the time to the receiver and half to the transmitter or in any predetermined fashion. Important applications can be found in infrared devices (IrDA) [16], [17], underwater acoustic communications [18], and in very high latency conditions (e.g., satellite and deep space communications) [19]–[21].

The key question to ask for achieving reliable communication in TDD channels is quite simple and natural: how much should a node transmit before stopping to listen to others' transmissions? This is particularly relevant in the presence of high latency (i.e., a large number of packets in flight) since the penalty for not transmitting (talking) the right amount of time before stopping is very high.

To provide an answer, we exploit synergies between (network) coding and feedback, where the latter is used to: 1) indicate the number of dofs missing at the receivers, rather than just signaling completion of the transmission as in previous studies, and 2) to tailor the redundancy of the code to the channel and system conditions.

In its simplest form, we consider a node transmitting M data packets to a single receiver using RLNC. The sender transmits RLNC packets back-to-back before stopping to wait for an acknowledgement (ACK) packet. This ACK indicates how many dofs are missing at the receiver to decode, say i. The number of coded packets N_i to be transmitted before stopping depends on i.

There is a natural tradeoff in the choice of the N_i's. If the N_i's are too small given the channel conditions, many ACK packets will be sent before completing a transmission, introducing unnecessary delay. If the N_i's are too large, the receiver may be unable to signal a successful transmission promptly. We show the existence of optimal N_i's and provide techniques to find good estimates for them.

By leveraging feedback and coding, our schemes are inherently hybrid. However, they are different from classical Hybrid ARQ (HARQ) schemes. HARQ focuses on exploiting soft combinations of previous transmissions of the same packet to enhance performance [22]. For example, an HARQ scheme will use different channel-code puncturings of the same packet at each transmission to provide new information to the receiver. For the erasure channel, such soft combining is not possible: a packet is either received or completely lost. Our schemes are also different from HARQ in that they: 1) rely on coding across packets, rather than on soft combinations of the same packet, for increased performance, and 2) an ACK feeds back precisely how much information is missing at the receiver rather than providing an estimate of it (e.g., stating successful decoding of the information (Type II HARQ [23]). As a final comment, HARQ would face similar challenges as ARQ or scheduling techniques in broadcast transmissions. Namely, a retransmission of the same packet (even with different code puncturing) will only benefit receivers missing that packet. On the other hand, coding across packets will allow a larger number of receivers to benefit from each RLNC packet transmission, as shown in [9].

II. Outline of Contributions

By tailoring feedback and coding, we show that it is possible to perform network coding in large latency TDD channels in an efficient manner. We study the benefits in terms of several metrics, such as mean completion time/energy and throughput. Section III begins by presenting preliminary results for the analysis of 1) absorbing Markov chains; 2) throughput in batch-by-batch schemes; and 3) probability distributions that will be used throughout this paper.

Our contribution can be separated into three main themes.

A. When to Stop Talking and Start Listening

We show that an optimal number of coded data packets exist to be transmitted back-to-back before stopping to wait for other nodes' transmissions, in terms of the mean completion time/energy, and provide techniques to find estimates for these values. The optimal N_i depends on the number of dofs i that the receiver requires to decode the information, on the packet erasure probability, and on the latency.

More specifically, Section IV studies the case of one node broadcasting information to several receivers (one-to-all broadcast). We prove that the computation of the optimal number of coded packets to transmit for the case of one receiver is simple and leverage it to propose heuristics that compute estimates for the case of multiple receivers. These heuristics have close-to-optimal performance while reducing the complexity of the estimation process. Section V extends our results to the case of all-to-all broadcast. We provide a full characterization of both cases by means of a moment generating function and compare to ARQ and scheduling schemes. Our results show that minimizing for the mean completion time under a TDD constraint yields close to or the same performance as a full-duplex system.

Although both standard ARQ techniques and our schemes achieve reliability by using feedback to recover lost packets, our schemes are different in that they rely on: 1) transmission of coded packets (i.e., there is no need to specify a particular data packet to retransmit); 2) ACKs that indicate the number of dofs needed at the receiver, rather than particular data packets, as in ARQ [23]; and 3) pre-emptive and adaptive redundancy determined by channel characteristics and feedback information. Previous work on rateless schemes typically assumes a dedicated, error-free channel to signal the completion of the data transfer. We assume a common transmission media for data and ACK, with a nonzero loss probability for the ACK. Other rateless codes (e.g., LT codes [24] or Raptor Codes [25]) will face similar challenges under the TDD channel. The advantage of RLNC is that an extension to general networks is simple and relies on re-encoding at intermediate nodes. End-to-end rateless erasure protecting codes (e.g., [24], [25]) do not share this trait.

The use of RLNC ensures that any M-coded packets received can be used to decode the original M data packets with high probability. This feature is similar to MDS coding schemes, such as Reed–Solomon codes, but with important differences. A Reed-Solomon code relies on a careful code design requiring:

1) *a priori* knowledge of the erasure probability in order to determine the required redundancy of the code and 2) all coded packets being generated at the same time before starting their transmission. RLNC has none of these constraints. Generating an RLNC coded packet is simple, on demand, and independent of previously encoded packets. An RLNC scheme can thus adapt easily to varying channel conditions. Finally, MDS codes will not allow for seamless re-encoding at intermediate nodes, which limits their impact in multihop network scenarios.

B. Practical Considerations

We address two key objections to the use of RLNC in practice. The first comes from the perception that very large field sizes are necessary in order to achieve good delay/throughput performance, which will compromise decoding complexity. The reasoning is that Galois fields of a larger size require more complex basic operations (e.g., multiplication) in order to code/decode. The second objection is that the decoding complexity of RLNC, which is $O(M^3)$ for decoding a batch of M data packets, is larger than for other rateless codes [24], [25]. For example, Raptor codes require $O(M\log(1/\epsilon))$ operations to recover the original data with $M(1+\epsilon)$ packets being received [25].

Section VI addresses these objections. First, we show that the use of a small field size causes very little degradation to performance, especially if the number of packets M to be combined is moderately large. More specifically, each receiver will need, on average, strictly less than $M+2$ coded packets to decode, regardless of the field size. Second, we propose the use of systematic RLNC as an approach that allows us to reduce decoding complexity and rely on small field sizes, while maintaining close to optimal throughput performance.

C. Random Arrival of Packets

In a more realistic network setting, packets are generated according to an arrival process. Thus, a sender's buffer may be empty or contain fewer than M packets awaiting transmission. A sender must choose to either wait for additional packets to arrive or take those packets in the buffer and start coding. We provide a joint characterization of this problem with our initial problem, namely, when to stop transmitting and start receiving.

The problem of queueing for network coding systems has been considered previously to account for burstiness or losses. This work is split between online approaches (e.g., [26]) and batch-by-batch approaches (e.g., [14], [15]), focusing on cases with slotted time. Our work is different in that it considers a TDD constraint, continuous time (not slotted), and a service time that depends on the number of packets being combined.

Section VII studies in detail an online network coding approach providing a nontrivial extension to the work in [26]. We then identify the required changes to capture batch-by-batch transmissions using the same general model.

Another important contribution of our work is to characterize the time between decoding events for online network coding. A decoding event constitutes decoding all packets that have been involved in linear combinations up to a given moment.

Conclusions are summarized in Section VIII.

III. Preliminaries

This section provides preliminary results in terms of figures of merit and general results relevant to our analysis, but useful also for problems with similar modeling and analysis techniques.

A. Absorbing Markov Chains

RLNC schemes for TDD channels in this paper can be accurately characterized using absorbing Markov chains. We prove general results for several relevant cases. We provide a full characterization of the cost associated with transitioning to the absorbing state via a moment generating function (MGF).

Definition: MGF of an Absorbing Markov chain: the MGF of a variable T associated with the transitions of an absorbing Markov chain starting in state S_l and ending in a unique absorbing state $\mathcal{A}$ is

$$M_{T,S_l}(s)=\sum_t \exp(st)P_T(T=t) \tag{1}$$

where $P_T(T=t)$ is the probability of $T=t$.

The following theorem states an iterative expression for the MGF of aperiodic, absorbing Markov chains.

Theorem 1: Let $M_{T,S_l}(s)$ be the MGF of an absorbing, aperiodic Markov chain where each state can only transition back to itself through self-transitions when starting in state S_l. Let us consider a cumulative random variable T with transition cost $T^{(S_j)}$ for each transition starting at state S_j. Then

$$M_{T,S_l}(s)=\frac{\exp(sT^{(S_l)})}{1-P_{S_l\to S_l}\exp(sT^{(S_l)})}\sum_{S_{l'}\neq S_l}P_{S_l\to S_{l'}}M_{T,S_{l'}}(s) \tag{2}$$

where $M_{T,\mathcal{A}}(s)=1$, $\mathcal{A}$ is the absorbing state, and $P_{S_l\to S_{l'}}$ is the probability of transitioning from state S_l to state $S_{l'}$.

Proof: Define $\mathcal{P}_{S_l}$ as a transition path to go from state S_l to the absorbing state $\mathcal{A}$ for the first time (i.e., $\mathcal{P}_{S_l}=\{S_l,S_l,S_{l_1},\ldots,S_{l_g},\mathcal{A}\}$). Define $T_{\mathcal{P}_{S_l}}$ as the cumulative cost (e.g., time or energy) for that transition path, and $\mathbf{P}(\mathcal{P}_{S_l})$ as the probability of choosing that path. Then, $M_{T,S_l}(s)=\sum_{\mathcal{P}_{S_l}}\exp\left(sT_{\mathcal{P}_{S_l}}\right)\mathbf{P}(\mathcal{P}_{S_l})$. By assumption, S_l can only be revisited by self-transitions and we can split $\mathcal{P}_{S_l}$ into a self-transition path with τ_{S_l} self-transitions and path $\mathcal{P}_{S_{l_1}}$ when a transition to a state $S_{l_1}\neq S_l$ occurs.

We define $P_{S_l\to S_{l_1}}$ as the probability to transition from S_l to S_{l_1}, and $\mathbf{P}(T_{S_l}=\tau_{S_l})$ as the probability of τ_{S_l} occurring before transitioning to S_{l_1}. Thus

$$\begin{aligned}M_{T,S_l}(s)&=\sum_{\tau_{S_l}}\sum_{S_{l_1}\neq S_l}\sum_{\mathcal{P}_{S_{l_1}}}\Big[\exp\left(sT_{\mathcal{P}_{S_{l_1}}}\right)P_{S_l\to S_{l_1}}\\&\cdot\mathbf{P}\left(\mathcal{P}_{S_{l_1}}\right)\exp\left(sT^{(S_l)}(\tau_{S_l}+1)\right)\mathbf{P}\left(T_{S_l}=\tau_{S_l}\right)\Big]\\&=\sum_{\tau_{S_l}}\exp\left(sT^{(S_l)}(\tau_{S_l}+1)\right)\mathbf{P}\left(T_{S_l}=\tau_{S_l}\right)\\&\cdot\sum_{S_{l_1}\neq S_l}P_{S_l\to S_{l_1}}M_{T,S_{l_1}}(s).\end{aligned} \tag{3}$$

We conclude the proof by showing that

$$\textstyle\sum_{\tau_{S_l}} \exp\left(sT^{(S_l)}(\tau_{S_l}+1)\right)\mathbf{P}\left(T_{S_l}=\tau_{S_l}\right) = \frac{\exp(sT^{(S_l)})}{1-P_{S_l\to S_l}\exp(sT^{(S_l)})}.$$

■

The following theorem states an extension to the Theorem 1 for periodic, absorbing Markov chains.

Theorem 2: Let $M_{T,S_{l,t_1}}(s)$ be the MGF of an absorbing, periodic Markov chain with period p in which each state can only transition back to itself through a transition path of the form $\{S_{l,t_2},\ldots,S_{l,t_p},S_{l,t_1}\}$ when starting in state S_{l,t_1}. If T is a cumulative random variable with transition cost $T^{(S_{j,t_i})}$ for each transition starting in state S_{j,t_i}, then

$$\begin{aligned} M_{T,S_{l,t_1}}(s) = {} & \frac{\exp(s\sum_{k=1}^{p}T^{(S_{l,t_k})})}{1-P_{S_l\to S_l}\exp(s\sum_{k=1}^{p}T^{(S_{l,t_k})})} \\ & \cdot\Big[\textstyle\sum_{S_{l',t_2}:l'\neq l}P_{S_{l,t_1}\to S_{l',t_2}}M_{T,S_{l',t_2}}(s) \\ & + P_{S_{l,t_1}\to S_{l,t_2}}\Big(\textstyle\sum_{a=2}^{p}\exp(s\sum_{k=2}^{a}T^{(S_{l,t_k})}) \\ & \cdot\textstyle\sum_{l'\neq l}F(a,l,l')M_{T,S_{l',t_{a+1}}}(s)\Big)\Big] \end{aligned} \tag{4}$$

where $M_{T,\mathcal{A}}(s)=1$, $\mathcal{A}$ is the absorbing state, $P_{S_{l,t_a}\to S_{l',t_b}}$ is the probability of transitioning from S_{l,t_a} to S_{l',t_b}, and

$$F(a,l,l') = \begin{cases} P_{S_{l,t_a}\to S_{l',t_{a+1}}} & \text{if } a=2,\\ P_{S_{l,t_p}\to S_{l',t_1}}\prod_{m=2}^{a-1}P_{S_{l,t_m}\to S_{l,t_{m+1}}} & \text{if } a=p,\\ P_{S_{l,t_a}\to S_{l',t_{a+1}}}\prod_{m=2}^{a-1}P_{S_{l,t_m}\to S_{l,t_{m+1}}} & \text{o.w.}\end{cases} \tag{5}$$

Proof: The proof technique is similar to the one used in Theorem 1. There are two main differences. First, the original transition path has several possible transitions from state S_{l,t_1} to itself going through states $\{S_{l,t_2},S_{l,t_3},\ldots,S_{l,t_p}$, which is similar to the case of self-transition in Theorem 1. Second, transitions that do not return to state S_{l,t_1} could constitute a) a direct transition from S_{l,t_1} to S_{l',t_2} with $l'\neq l$ or b) a transition from S_{l,t_1} to S_{l,t_2} and from there possibly to other states S_{l,t_k} until a transition occurs from S_{l,t_k}, $2\leq k\leq p$ to a state of the form $S_{l',t_{k'}}$ for $l'\neq l$, and $k'=k+1$ if $k<p$, and $k'=1$ if $k=p$. ■

The mean of a variable T associated with the cost of transitions in an absorbing Markov chain before the system reaches $\mathcal{A}$ and when the system is in state S_l is given by

$$T_{S_l}=T^{S_l}+\sum_{S_l,S_{l'}}P_{S_l\to S_{l'}}T_{S_{l'}}. \tag{6}$$

We can express this in vector form as

$$\bar{T}=(I-\wp)^{-1}\bar{\mu} \tag{7}$$

where $\bar{T}=[T_{S_l}]$, $\bar{\mu}=[T^{(S_l)}]$, $\wp$ is the corresponding transition probability matrix, and $[a_i]$ represents the vector formed by elements a_i, $\forall\ i$. If we are interested in the mean cost when we start at state S_0, we can use Cramer's rule to determine

$$T_{S_0}=\frac{\det\left(\Gamma\leftarrow_{S_0}\bar{\mu}\right)}{\det\left(\Gamma\right)} \tag{8}$$

where $\Gamma=I-\wp$, and $\Gamma\leftarrow_{S_0}\bar{\mu}$ represents a matrix that has all columns as the Γ matrix except the column corresponding to state S_0 which is substituted by the vector $\bar{\mu}$.

B. Performance Metrics

Let us now define some performance metrics to be used in our discussion of block transmission schemes.

Definition: Completion time (energy): of a block-by-block scheme constitutes the time (energy) required to transmit a block of data packets reliably to the intended receivers and receive confirmation of successful delivery of the block of packets. If the process is characterized by an absorbing Markov chain, the completion time (energy) constitutes the accumulated time (energy) from transitions starting at an initial state and until the first transition to an absorbing state.

Definition: Mean throughput of a block scheme: The mean throughput for a block scheme is strictly defined as

$$\text{Mean Throughput}=E[\frac{Mn}{T}] \tag{9}$$

where T is the time to complete transmission of M packets, each containing n bits of information.

If we assume M and n to be constants, which is valid in our scheme, we have

$$\text{Mean Throughput}=MnE[\frac{1}{T}]=Mn\int_0^\infty M_{T,S_l}(-s)ds \tag{10}$$

where the second term comes from the expression of negative moments [27], [28].

Note that $M_{T,S_l}(-s)$, $\forall\ S_l$ proven in Theorems 1 and 2 have a left-most multiplying term which decreases to zero exponentially as $s\to\infty$ for $T^{(S_l)})>0$. Thus, all terms inside the integral in (10) will go to zero exponentially. We can determine $E[T^{-1}]$ by using numerical integration techniques and the following approximation:

$$E[T^{-1}]\approx\int_0^\tau M_{T,S_l}(-s)ds \tag{11}$$

where $\tau=\max_{\{i=1,\ldots,j\}}\tau_i$, $\tau_i=C/T^i$, and C is a constant in order to ensure $\exp(-\tau_iT^i)$ is small enough (e.g., $C=5$ ensures $\exp(-\tau_iT^i)=\exp(-5)\approx 0.0067$).

Although the mean throughput is important, we define a different throughput measure called η because 1) the mean throughput is computationally demanding, and 2) most of the analysis of typical ARQ schemes is performed using η.

Definition: Throughput measure η for block-by-block transmissions: throughput measure η constitutes the ratio between the number of data bits transmitted (n) and the time it takes to transmit them. For the case of a block-by-block transmission

$$\eta=\frac{Mn}{E[T]} \tag{12}$$

where $E[T]$ is the mean time to complete the transmission of the data.

Using Jensen's inequality, it is clear that η constitutes a lower bound to the mean throughput. Finally, minimizing the mean time to complete transmission of a block of M data packets with n bits each is equivalent to maximizing η for those values.

C. Extended Binomial Distribution

The extended binomial distribution is used in many of our discussions and is expressed as follows:

$$\mathbf{P}_{(i,N,p)}(k) = \begin{cases} \binom{N}{k}\left(\frac{1-p}{p}\right)^k p^N, & \text{if } 0 \le k < i, N \ge i \\ \sum_{a=i}^{N} \binom{N}{a}\left(\frac{1-p}{p}\right)^a p^N, & \text{if } k = i, N \ge i \\ 0, & \text{otherwise} \end{cases}$$

where $1-p$ is the probability of an event A (e.g., a packet was successfully received) and k is the number of A events that occur in N trials. Finally, i indicates the maximal number of A events that can be accumulated in a meaningful fashion (e.g., receiving 10 or 7 boxes into a storage warehouse that can only fit 7 boxes causes the same end result, namely, the warehouse is full). Thus, the case $k = i$ concentrates the probability of i or more A events occurring.

IV. Random Linear Network Coding for One-to-All Broadcast in TDD Channels

We now analyze the problem of one-to-all broadcast and provide simple, useful heuristics to estimate the number of coded data packets to be transmitted. We compare the proposed schemes to optimal scheduling policies.

A. Model

A sender wants to broadcast M data packets at a given data rate R [bps] to N receivers. We assume an independent packet erasure channel from the sender to each of the receivers, where Pe_j and Pe_{ack_j} represents the erasure probability of a coded packet and of an ACK packet for receiver j, respectively. We assume only single-hop transmissions and no cooperation amongst receivers. Nodes have a TDD constraint. Each transmitted RLNC coded packet contains a linear combination of the M data packets of n bits each, as well as the random coding coefficients used in the linear combination. Each coefficient is represented by g bits. For encoding over a field size q, we have $g = \log_2 q$ bits. Also, consider an information header of size h. Thus, the total number of bits per packet is $h + n + gM$. The transmission time of a coded packet is $T_p = \frac{h+n+gM}{R}$. T_{ack} constitutes the transmission time of each ACK packet, where $T_{\text{ack}} = n_{\text{ack}}/R$, and n_{ack} is the number of bits in the ACK packet. We define T_{rt_j} as the round trip time to receiver j and $\mathcal{E}_p$ and $\mathcal{E}_{\text{ack}}$ represent the energy per coded and ACK packet, respectively. Note that connectivity amongst receivers is not specified, thus allowing for any number of topologies to be mapped into the current single-hop broadcast scenario. The only requirement is that the connection between each receiver j and the sender is characterized by $(Pe_j, Pe_{\text{ack}_j}, T_{rt_j})$.

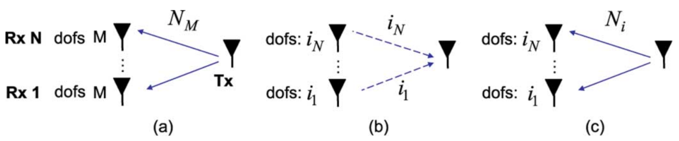

Fig. 1. Suboptimal selection of coded packets in our network coding TDD scheme for one-to-all broadcast. (a) The transmitter initially generates N_M-coded packets from the M packets in its queue and sends them to the receivers before stopping to wait for the ACK packets. (b) Each receiver k send an ACK packet indicating that i_k dofs are needed to decode. (c) Upon reception of the ACK packet, the transmitter updates its knowledge of the receivers' requirements and generates N_i-coded packets, where $i = \max_{k=1,2,\ldots,N} i_k$, and sends them to the receivers.

We assume that the field size q is large enough so that the expected number of successfully received packets at the receiver, in order to decode the original data packets, is approximately M [9]. This is not a necessary assumption for our analysis, as we show in Section VI.

B. Description of Time-Division Duplexing Scheme

The sender can transmit coded packets back-to-back before stopping to wait for an ACK packet from each receiver. Each ACK feeds back the number of dofs that are still required to decode the M data packets at a given receiver.

The process is modeled as an absorbing Markov chain. The states $(s_1, s_2, \ldots, s_N)$ are defined by the number of dofs required s_k at receiver k to successfully decode the M packets. Thus, the states range from $(M, M, \ldots, M)$ to $(0, 0, \ldots, 0)$. This is a Markov chain with $(M+1)^N - 1$ transient states and one recurrent state (state $(0, 0, \ldots, 0)$).

The optimal scheme would require us to associate one variable, representing the number of coded packets to be transmitted, to each state in our Markov chain (i.e., $N_{(s_1,s_2,\ldots,s_N)}$-coded packets should be transmitted if the system is in state $(s_1, s_2, \ldots, s_N)$. Due to complexity, we propose a suboptimal scheme that requires only M variables to be optimized.

Thus, we consider that the transmitter sends N_i coded packets, where $i = \max_{j=1,2,\ldots,N} i_j$. If an ACK is lost, the transmitter assumes the previous state for the corresponding receiver. The communication process is illustrated in Fig. 1. At the beginning, each receiver requires M dofs to decode the information. The transmitter starts by sending N_M-coded packets before stopping to listen for ACKs [Fig. 1(a)]. Each receiver then sends an ACK packet indicating how many dofs it requires to decode, say $i_1, i_2, \ldots, i_N$ for receivers $1, 2, \ldots, N$, respectively [Fig. 1(b)]. Then, the transmitter sends N_i-coded packets before stopping [Fig. 1(c)]. This process is repeated until all M packets have been decoded successfully by all receivers.

We expect the choice of N_i to minimize the completion time for: 1) the case of one channel has a much larger erasure probability as the others, since it constitutes the bottleneck of the system, and 2) very high latency channels, because our scheme will aim to ensure completion after the first transmission, thus making N_M the most important variable to determine system performance.

The transition probabilities from state $(s_1, s_2, \ldots, s_N)$ to state $(s'_1, s'_2, \ldots, s'_N)$ are

$$P_{(s_1,s_2,\ldots,s_N)\to(s'_1,s'_2,\ldots,s'_N)} = P\left(X_1(n)=s'_1,\ldots,X_N(n)=s'_N | X_1(n-1)=s_1,\ldots,X_N(n-1)=s_N\right)$$

where $X_i(n)$ is the number of dofs required at receiver i at the end of transmission n. For simplicity of notation, let us say that $P\left(X_1(n)=s'_1,\ldots,X_N(n)=s'_N|X_1(n-1)=s_1,\ldots,X_N(n-1)=s_N\right)=P\left(s'_1,\ldots,s'_N|s_1,\ldots,s_N\right)$ and that $P\left(s'_i|s_i,\max_{j=1,2,\ldots,N} s_j\right) = P\left(X_i(n)=s'_i|X_i(n-1)=s_i,\max_{j=1,2,\ldots,N} s_j\right)$. If we consider independent packet erasure channels for each of the receivers and that the dependence on the previous state $(s_1, s_2, \ldots, s_N)$ can be translated into a dependence on the state with maximum number of required dofs (i.e., $i = \max_{j=1,2,\ldots,N} s_j$), because i determines N_i, then

$$P_{(s_1,s_2,\ldots,s_N)\to(s'_1,s'_2,\ldots,s'_N)} = \prod_{m=1}^{N} P\left(s'_m|s_m,N_i\right). \quad (13)$$

For $0 \leq s'_j < s_j$, this can be translated into $P\left(s'_j|s_j,N_i\right) = (1 - Pe_{\text{ack}_j})\mathbf{P}_{(s_j,N_i,Pe_j)}(s_j - s'_j)$. For $s_j = s'_j > 0$, the expression for the transition probability reduces to $P\left(s_j|s_j,N_i\right) = (1 - Pe_{\text{ack}_j})\mathbf{P}_{(s_j,N_i,Pe_j)}(0) + Pe_{\text{ack}_j}$. Finally, $P\left(0|0,N_i\right) = 1$.

C. Completion Time

The completion time is determined by the time to absorption of the Markov chain. The transition time each state $(s_1, \ldots, s_N)$ with $i = \max_j s_j$ is T^i, which corresponds to the time to transmit N_i RLNC packets and receive ACKs from all receivers (i.e., $T^i = N_i T_p + T_w$). The waiting time between stopping transmission and reception of the ACKs is T_w. Reference [29] provides more details as to how to coordinate transmissions of the ACKs to reduce T_w. If $T_{rt} = T_{rt_j}, \forall\, j$, then $T_w = T_{rt} + NT_{\text{ack}}$.

Remark: The MGF of the completion time for one-to-all broadcast is of the form of Theorem 1.

Only small modifications are needed to characterize the problem of completion energy, namely $\mathcal{E}^{(s_1,\ldots,s_N)} = N_{(\max_i s_i)}\mathcal{E}_p + \mathcal{E}_{\text{ack}}$ will represent the transition cost at state $(s_1, \ldots, s_N)$.

D. Estimating the Number of Coded Packets to Transmit

Let us first consider the case of a single receiver. We exploit the structure of the problem to provide a simple, iterative optimization mechanism. Theorem 3 summarizes this mechanism and shows it is optimal.

Theorem 3: Let $N_i*, i = 1, \ldots, M$ be the optimal values of N_i for our RLNC TDD scheme with a single receiver in terms of the completion time (T_M). Finding N_i* can be determined by first optimizing $N_j*, \forall\, 1 < j < i$. Finding $N_i*, i = 1, \ldots, M$ can be solved iteratively by M one-variable searches. The optimal value N_1* depends only on system parameters.

Proof: The mean completion time when the system is in state i is given by $T_i = T^i + \sum_{j\leq i} P_{i\to j}T_j$. Since $T^i = N_i T_p + T_w$, and the transition probabilities of each state i depend only on N_i, then for any j

$$\min_{N_j,\ldots,N_1} T_j = \min_{N_j}\left(\frac{T^j + \sum_{i=1}^{j-1} P_{j\to i}\min_{N_i,\ldots,N_1} T_i}{1-P_{j\to j}}\right).$$

Thus, the problem can be solved iteratively by first computing $\min_{N_1} T_1$ and then using this result to find $\min_{N_2,N_1} T_2$, and so on. ■

The problem of one receiver has a simple solution. However, the structure of the problem for $N > 1$ receivers does not yield such a simple optimization mechanism. Thus, we consider a simple heuristic to estimate the values of $N_i*, \forall\, i = 1, \ldots, M$.

Our heuristic, named the 'Worst Link Channel' heuristic, approximates the system as a link to the receiver with the worst channel (i.e., $Pe = \max_j Pe_j$). Then, we compute $N_i, \forall\, i = 1, \ldots, M$ to minimize the mean completion time with the network coding scheme studied for a link using the current values of T_p, T_w, and $Pe_{\text{ack}} = \max_j Pe_{\text{ack}_j}$.

Remark: The N_i's do not need to be computed in real time. They can be precomputed for different channel conditions (e.g., Pe_j, T_{rt}, and/or system settings, for example, n, M, g, R) and stored in the receiver as lookup tables, thus making the computational load on the nodes negligible during normal operation.

E. Performance Analysis and Numerical Results

This subsection provides numerical examples and comparison schemes for our proposed TDD network coding scheme. We consider first some comparison schemes for the case of a single receiver and then for the case of multiple receivers.

1) Link: We focus on large latency cases inspired by constraints of satellite communications, although examples with other large latency cases (e.g., underwater communications) could also be analyzed.

2) Network Coding for TDD Optimized for Mean Completion Time (TDD-T): This is our TDD scheme when we choose the N_i's to optimize the mean completion time given channel characteristics and system parameters.

3) Network Coding for TDD Optimized for Mean Completion Energy (TDD-E): This is our TDD scheme when we choose the N_i's to optimize the mean completion energy given channel characteristics and system parameters.

4) Network Coding in Full-Duplex Channel: This scheme assumes that nodes can receive and transmit simultaneously. The sender transmits coded packets back-to-back to the destination. Once the M packets have been decoded, the receiver transmits ACK packets back-to-back, each of duration T_{ack}. The sender keeps transmitting until an ACK packet for correct decoding of all information has been received. This scheme is optimal in light of minimal delay and shall be used for illustrating the gap of other schemes to the best achievable completion time for a link. This scheme can be modeled as a Markov chain where, as before, the states represent the number of dofs received. The time spent in each state is the same (T_p). The mean time to complete the transmission and get an ACK is

$$E[T] = T_{rt} + \frac{MT_p}{1 - Pe} + \frac{T_{\text{ack}}}{1 - Pe_{\text{ack}}} \quad (14)$$

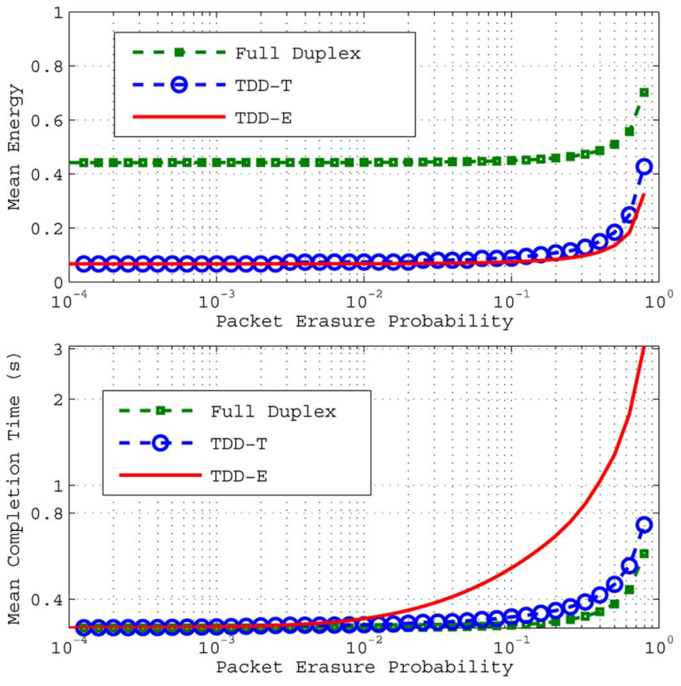

Fig. 2. Mean energy and time to complete transmission. Parameters used: $M = 10$, packet size $n = 10{,}000$ bits, $R = 1.5$ Mb/s, $h = 80$ bits, $g = 20$ bits, $n_{\text{ack}} = 100$ bits.

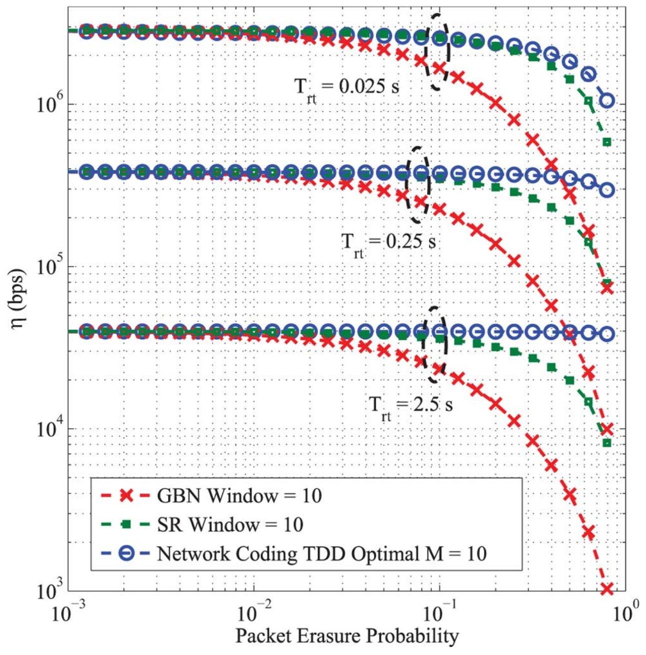

Fig. 3. η versus data packet erasure probability with two TDD non-network coding schemes (GBN and SR) and our TDD-T scheme, with different T_{rt} values. We used as parameters $g = 20$ bits, $n_{\text{ack}} = 100$ bits, $n = 10000$ bits, $h = 80$ bits, $R = 10$ Mb/s, $Pe_{ACK} = 0$.

where T is the time to complete transmission of M packets. The mean energy to complete the transmission and get an ACK is

$$E[\text{Energy}] = \frac{T_{rt}\mathcal{E}_p}{T_p} + \frac{T_{rt}\mathcal{E}_{\text{ack}}}{2T_{\text{ack}}} + \frac{M\mathcal{E}_p}{1-Pe} + \frac{\mathcal{E}_{\text{ack}}}{1-Pe_{\text{ack}}}. \tag{15}$$

5) Go-Back-N ARQ for TDD (GBN): This is an ARQ scheme developed for a TDD duplex channel in [16]. Each transmission contains W data packets sent back-to-back, where W is the window size. The η for this scheme is given by

$$\eta_{\text{GBN}} = \frac{n(1-Pe)\left(1-(1-Pe)^W\right)}{(WT_p + T_w)Pe}. \tag{16}$$

6) Selective Repeat ARQ for TDD (SR): This is an ARQ scheme developed for a TDD duplex channel in [16]. Each transmission contains W data packets, where W is the window size. Using results from [16], we provide the equivalent η for this problem

$$\eta_{SR} = \frac{Wn(1-Pe)}{WT_p + T_w}. \tag{17}$$

Let us now present some numerical results comparing these schemes. Fig. 2 studies the mean completion time and energy of $M = 10$ 10 data packets with different Pe in a GEO satellite link with a propagation delay of 125 ms. We have considered that coded packets and ACK are transmitted with the same power, and that this value is normalized (i.e., $Po = 1$). The remaining parameters are specified in the figure. TDD-T and the network coding full-duplex optimal scheme have similar performance over a wide range of block erasure probabilities. For the worst case $(Pe = 0.8)$, TDD-T requires only 29% more time to complete than the full-duplex scheme. This is surprising as the transmitter in the full-duplex scheme sends coded packets nonstop until an ACK packet is received. The explanation for this behavior is that our scheme is sending enough coded packets, given the channel conditions, so that the number of stops to listen is minimized.

Fig. 2 shows that TDD-T and TDD-E have much better performance with respect to the full-duplex scheme (i.e., energy consumption of the full-duplex scheme is considerably higher than the TDD schemes given the high latency characteristic of this channel).

Fig. 2 shows that the performance of TDD-T and TDD-E remains similar over a wide range of Pe. When Pe is low, the performance is the same both in energy and delay. For high Pe, the performance of both TDD versions is similar in terms of energy, although we observe a clear advantage of TDD-T over TDD-E in mean completion time.

Remark: Our TDD-T scheme provides a good tradeoff between energy and time to complete transmissions.

Let us now compare the throughput performance of our scheme to that of TDD ARQ schemes 4 and 5. Fig. 3 shows η for a fixed data rate of 10 Mb/s and different T_{rt}. We use the parameters in the figure, with a window size of $W = 10$ for the ARQ schemes and $M = 10$ for TDD-T. The performance of our scheme is the same as both GBN and SR at low Pe. Since the data rate is kept fixed, at higher T_{rt}, we get higher latency. For low latency, η of our scheme is very close to that of the SR ARQ scheme for all values of Pe, and better than the GBN scheme for high Pe. As latency increases, our scheme shows much better performance than the SR scheme for high Pe. The case of $T_{rt} = 2.5$ s and $Pe = 0.8$ shows that η of our scheme is more than five times greater than that of SR. These results are surprising since our scheme relies on completely transmitting one batch of M packets before going to the next batch, which is not the case for GBN or SR.

Remark: TDD-T transmits reliably one block of M data packets before transmitting a new one, guaranteeing in-order delivery of the original packets. In contrast, SR does not provide such guarantee of delay for any single data packet, which is unacceptable in many applications. Thus, our comparison to SR is not fair, as it favors SR. Nonetheless, our scheme provides similar or better throughput than SR. The comparison is based under similar ACK signaling strategies.

7) One-to-All Broadcast: We first extend the work in [9] to determine the mean completion time for optimal scheduling policies for one-to-all broadcast in order to compare them to our RLNC scheme, which we identify as broadcast TDD. These policies consider no coding of the data packets, no channel-state information, and nodes that only send ACK when they have received all M data packets.

8) Broadcast With Round Robin in Full-Duplex Channel (RR Full Duplex): Since the channels are independent and identically distributed over time and users, Round Robin (RR) constitutes an optimal policy. Thus, packet k in the block is transmitted every $(mM+k)T_p$ time units for $m=0,1,2,\ldots$ until all the receivers get all M packets [9]. Using a similar analysis as in [9]

$$E[T] = T_w + T_p M\left(\gamma + E[\max_{i,k} X_k^i]\right) \tag{18}$$

where $1+X_k^i$ is the number of transmissions of packet k needed to reach node i, $\gamma \in (1/2, 1)$, and

$$E[\max_{i,k} X_k^i] = \sum_{t=1}^{\infty}\left[1-(1-Pe^t)^{\mathrm{MN}}\right] \tag{19}$$

where $Pe = Pe_1 = \cdots = Pe_N$. An upper and lower bound on $E[T]$ is given by $\gamma = 1$ and $\gamma = 1/2$, respectively.

9) Broadcast With Round Robin in TDD (RR TDD): This scheme assumes limited feedback due to the TDD constraint. The transmitter broadcasts all M packets back-to-back, then stops to receive ACK packets that indicate completion of the entire file. If there are nodes that have not acknowledged the block of packets, the transmitter repeats the process (i.e., sends all M packets and stops to listen for ACKs). The mean completion time of this scheme is

$$E[T] = (T_w + T_p M)\, E[\max_{i,k} X_k^i]. \tag{20}$$

We now provide numerical examples that compare our network coding scheme to these scheduling schemes in a satellite example. For simplicity, we consider that there are no erasures of ACK packets and that the distance between the transmitter and each receiver is the same. The latter is a good approximation in many satellite scenarios. We compare our broadcast scheme with RR TDD and RR Full Duplex.

Fig. 4 compares the Broadcast TDD scheme with N_i's computed optimally and with the 'Worst Link Channel' heuristics, and compares it to the performance of RR TDD and RR Full Duplex. For the range of packet erasures considered, our coding scheme performs at least as well as the RR TDD, and considerably better at high erasures. Broadcast TDD has similar performance to RR Full Duplex for low erasures. However, our coding scheme performs better at high packet erasures ($Pe_1 = Pe_2 > 0.3$). In fact, at $Pe_1 = Pe_2 = 0.8$, RR Full Duplex takes 20% more time to complete transmissions.

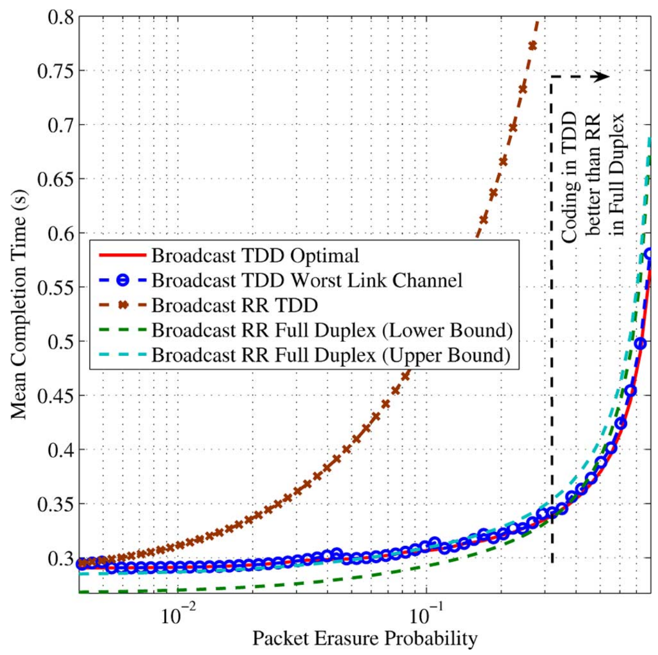

Fig. 4. Mean completion time for Broadcast TDD with the optimal choice of N_i's, Broadcast TDD with the "Worst Link Channel" heuristic, RR Full duplex, and RR TDD. We use as parameters $N = 2$ receivers at the same distance from the transmitter $M = 5$, $Pe = Pe_1 = Pe_2$, $R = 1.5$ Mb/s, $h = 80$ bits, $g = 20$ bits, $n_{\text{ack}} = 50$ bits.

Remark: Broadcast TDD for one-to-all broadcast performs better than RR Full Duplex for an important range of Pe by tailoring coding and feedback appropriately. Any RR TDD scheduling scheme will not outperform RR Full Duplex because: 1) the feedback shares the channel with the data and 2) feedback is sent sporadically, delaying the time to stop transmission.

V. Random Linear Network Coding for All-to-All Broadcast in TDD Channels

Section IV considered the case in which only one node has information to transmit. We now analyze the problem of networks of N nodes in which all nodes want to share disjoint information on a single TDD channel. Although we focus on completion time, the analysis extends easily for completion energy. We provide a simple algorithm to determine the number of coded data packets to be transmitted back-to-back before stopping to allow transmission from other nodes.

A. Model

Each node i has M_i data packets to share with all other nodes. Nodes transmit following a round robin assignment, where the order of transmission has been predefined. Node i can transmit at a rate R_i (b/s). We assume an independent, memoryless packet erasure channel with a probability of erasure for transmissions from node i to node j, $Pe_{i,j}$. Any transmission from a node can be received by all other nodes. Finally, we assume that the next node to transmit waits for all nodes to receive the previous transmission. T_{prop_i} is the propagation time from node i to the node that is farthest from it.

Each node is both a sender and a receiver. When node i operates as the sender, it transmits RLNC coded data packets of its M_i data packets back-to-back before stopping. The ACKs to node i come piggybacked in the header of each coded packet sent from the other nodes and are assumed to suffer no erasures. We consider a large q.

The process is modeled as an absorbing, periodic Markov chain where a transition occurs every time that a batch of back-to-back coded packets is received by all nodes. The vector $S_a = (s_{a,1}, \ldots, s_{a,N})$ represents the dofs that other nodes require from node a in order to decode its information (i.e., $s_{a,b}$ constitutes the dofs required by node b to decode all packets from a). We define $S = (S_1, S_2, \ldots, S_N)$, so that (S,t) represents a state of the Markov chain with the node t acting as a transmitter in this state. There are $N(M_1+1)^{N-1}(M_2+1)^{N-1}\cdots(M_N+1)^{N-1} - N$ transient states and N recurrent states.

An optimal solution requires a variable per state (i.e., $N_{((S_1,S_2,\ldots,S_N),t)}$, $\forall\ S_i,\ t$), which increases exponentially with N. Since this becomes computationally infeasible, we reduce the number of variables to optimize to $N(M_1+1)\ldots(M_N+1) - N$ by considering only the maximum dofs $s_a = \max_b s_{a,b}$ that the receivers of node a need. We rename the variables to optimize as $N_{((s_1,s_2,\ldots,s_N),t)}$. The transition probabilities from state (S,t) to state (S',t') are given by

$$P_{(S,t)\to(S',t')} = \begin{cases} P\left(S'_a|S_a, N_{((s_1,\ldots,s_N),t)}\right), & \text{if } S_b = S_b', \forall\, b \neq a, \\ & t = a, t = t_{next}(a), \forall\, a \\ 0, & \text{o.w.} \end{cases}$$

where $P\left(S'_a|S_a,N_{((s_1,\ldots,s_N),t)}\right)$ is the probability of transitioning from S_a to S'_a when node a has transmitted $N_{((s_1,\ldots,s_N),t)}$-coded packets, and $t_{\text{next}}(a)$ represents the next node that should transmit after node a has transmitted. For independent channels

$$P\left(S'_a|S_a,N_{((s_1,\ldots,s_N),t=a)}\right) = \prod_{j\neq a} P\left(s'_{a,j}|s_{a,j},N_{((s_1,\ldots,s_N),t=a)}\right)$$

where $P\left(s'_{a,j}|s_{a,j},N_{((s_1,\ldots,s_N),t)}\right)$ represents the transition probability related to the knowledge of node j with respect to the data of a, when node a sends $N_{((s_1,\ldots,s_N),t)}$-coded packets. For ease of notation, we substitute $N_{((s_1,\ldots,s_N),t)}$ for Nt. For $0 \leq s_{a,j'} \geq s_{a,j}$, $P\left(s'_{a,j}|s_{a,j},Nt\right) = \mathbf{P}_{(s_{a,j},Nt,Pe_{a,j})}(s_{a,j} - s_{a,j'})$. Finally, $P\left(0|0,Nt\right) = 1$.

B. Completion Time

The time for completing the sharing process of all data packets to all of the nodes constitutes the time of absorption, that is, the time to reach any state $((0,\ldots 0),\ldots,(0,\ldots 0),t)$ for some t for the first time, given that the initial state is $((M_1,\ldots,M_1),\ldots,(M_N,\ldots,M_N),t')$ for some t' which starts the transmission process. We consider that the transmission time of a packet from node i is given by $T_{p_i} = \frac{h+n+gM_i}{R_i}$. We define $T^{(S,t)}$ as the time it takes to transmit $N_{(S,t)}$ coded data packets and reach the node that is farthest away from t, that is, $T^{(S,t)} = N_{(S,t)}T_{p_t} + T_{\text{prop}_t}$, and $T_{(S,t)}$ as the mean completion time given starting state (S,t).

Remark: The MGF of the completion time for all-to-all broadcast is of the form of Theorem 2.

The optimization is computationally prohibitive even after reducing the number of variables to optimize. We consider that the round-trip time that depends on the physical round-trip time and the transmissions of other nodes in the system, by observing that

$$\begin{aligned} T_{(S,t_1)} &= N_{(S,t_1)}T_{p_{t_1}} + \sum\nolimits_{b=1}^{N}\left(T_{\text{prop}_{t_b}}\right) \\ &+ \sum\nolimits_{n=2}^{N} T_{p_{t_n}} E_{t_2,\ldots,t_n}\left[N_{(i_1,\ldots,i_N,t_n)}|(S,t_1)\right] \\ &+ \sum\nolimits_{S',\ldots,S^{(N)}} T_{(S',t_1)}\mathbf{P}_{(S,t_1)\to(S',t_1)} \end{aligned} \tag{21}$$

where

$$\begin{aligned} \mathbf{P}_{(S,t_1)\to(S',t_1)} = &\sum_{S^{(2)},\ldots,S^{(N)}} \Big[P_{(S^{(N)},t_N)\to(S',t_1)} \\ &\cdot \prod_{i=1}^{N-1} P_{(S^{(i)},t_i)\to(S^{(i+1)},t_{i+1})} P[S^{(2)},\ldots,S^{(N)}]\Big] \end{aligned} \tag{22}$$

and $P[S^{(2)},\ldots,S^{(N)}]$ constitutes the probability of transitioning from (S,t_1) to (S',t_1) through the path given by states $(S^{(2)},t_2),\ldots,(S^{(N)},t_N)$, and $S = S^{(1)}$. Also

$$\begin{aligned} E_{t_2,\ldots,t_n}&\left[N_{(i_1,\ldots,i_N,t_n)}|(S,t_1)\right] = \\ &\sum_{S^{(2)},\ldots,S^{(n)}} N_{(S^{(n)},t_n)}\mathbf{P}_{(S,t_1)\to(S^{(n)},t_n)} \end{aligned} \tag{23}$$

where

$$\begin{aligned} &\mathbf{P}_{(S,t_1)\to(S^{(n)},t_n)} = \\ &\sum_{S^{(2)},\ldots,S^{(n-1)}} \prod_{i=1}^{n} P_{(S^{(i-1)},t_{i-1})\to(S^{(i)},t_i)} P[S^{(2)},\ldots,S^{(n-1)}]. \end{aligned}$$

We propose an algorithm that leverages this structure and the one-to-all broadcast heuristics to obtain good estimates. The key intuition behind the algorithm is that each node perceives the problem as a one-to-all broadcast problem, where the effect of other nodes' transmissions are coupled in the random waiting time of each one-to-all broadcast problem.

Let us define $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n)$ as the estimate for $N_{(s_1,s_2,\ldots,s_N,t_k)}$ at step n of the algorithm. Then, our algorithm can be written as follows.

Algorithm 1 Search Algorithm for N nodes.

```
STEP 0 : INITIALIZE
-Set $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(0) = s_k, \forall\, k$, and $n = 1$.
STEP 1 : TRANSMISSION FROM NODE 1 :
-Set $\hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n) = \hat{N}_{(s_1,s_2,\ldots,s_N,t_k)}(n-1), \forall\, k$.
FOR $s'_2 = 1, 2, \ldots, M_2$
...
FOR $s'_N = 1, 2, \ldots, M_N$
```

```
    Compute $\hat{N}_{(s_1,s'_2,\ldots,s'_N,t_1)}(n)$, $\forall\ s_1 = 1,\ldots,M_1$ to
    minimize the completion time of a TDD link with
    $Pe = \max_b Pe_{1,b}$, and with transition cost
```

$$\hat{N}_{(s_1,s'_2,\ldots,s'_N,t_1)}(n)T_{p_1} + \sum_{b=1}^{N}\left(T_{\mathrm{prop}_{t_b}}\right) + \sum_{y=2,\ldots,N} T_{p_{t_y}} E_{t_2,\ldots,t_y}\left[\hat{N}_{(i_1,\ldots,i_N,t_y)}(n)|(S,t_1)\right] \quad (24)$$

```
    where $S = (s_1, s'_2, \ldots, s'_N)$.
    END FOR
  ...
  END FOR
STEP k (= $2,\ldots,N$): TRANSMISSION FROM NODE k TO NEXT NODE:
  FOR $s'_1 = 1,2,\ldots,M_1$
  ...
  FOR $s'_N = 1,2,\ldots,M_N$
    Compute $\hat{N}_{(s'_1,\ldots,s_k,\ldots,s'_N,t_k)}(n)$, $\forall\ s_k = 1,\ldots,M_k$
    to minimize the completion time of a TDD link with
    $Pe = \max_b Pe_{k,b}$, and with transition cost
```

$$\hat{N}_{(s'_1,\ldots,s'_N,t_k)}(n)T_{p_k} + \sum_{b=1}^{N}\left(T_{\mathrm{prop}_{t_b}}\right) + \sum_{y=k+1,\ldots,N,1,\ldots k-1} T_{p_{t_y}} E_{t_k,\ldots,t_y}\left[\hat{N}_{(i_1,\ldots,t_y)}(n)|(S,t_k)\right] \quad (25)$$

```
    where $S = (s'_1,\ldots,s_k,\ldots,s'_N)$.
  END FOR
  ...
  END FOR
STEP N + 1 : STOPPING CRITERIA
  IF $\hat{N}_{(s_1,\ldots,s_N,t)}(n) = \hat{N}_{(s_1,\ldots,s_N,t)}(n)(n-1)$,
  $\forall\ s_1,\ldots,s_N,t$, STOP
  ELSE $n = n+1$, and go to Step 1.
  END IF
```

C. Performance Analysis and Numerical Results

We now compare our TDD RLNC scheme to: 1) a full-duplex RLNC scheme and 2) scheduling schemes for TDD and full-duplex channels. We focus on the case of two nodes to provide simple, optimal scheduling policies. The scheduling policies consider no coding across data packets, no channel state information, and nodes that only ACK when they have received all information. We analyze the different schemes and provide numerical results to illustrate the advantages of our approach. All schemes piggyback the ACK to the header of each transmitted coded packet. As in [9], we restrict the analysis to independent symmetric channels (i.e., $Pe_1 = Pe_2$ and $T_{p_0} = T_{p_1} = T_p$) and no erasures in the ACKs for tractability of the scheduling schemes.

1) Data Sharing With Network Coding in the Full-Duplex Channel (DSNC Full Duplex): Each node generates random linear combinations of its original M_i data packets, and sends those coded packets back-to-back through the channel to the other node. We assume that both nodes start transmitting at the same time to reduce the completion time of the sharing process. This problem can be modeled through a Markov chain with states (i,j), where i and j represent the dofs required to decode at nodes 1 and 2, respectively. Since both nodes start transmitting at the same time, and we assume the packets take the same time to be transmitted T_p, then transitions occur every arrival of a coded packet. The transition probabilities are modeled as $P_{(i,j)\to(i',j')} = P_{i\to i'}P_{j\to j'}$ where we assume independence of the channels, $P_{0\to 0} = 1$ and $P_{i\to i'} = \mathbf{P}_{(1,1,Pe)}(i-i')$. Using techniques from Section III, the mean completion time is

$$E[T] = T_w + T_p \frac{\det\left(\Gamma \leftarrow_{(M,M)} \bar{1}\right)}{\det(\Gamma)} \quad (26)$$

where $T_w = 2T_{\mathrm{prop}} + T_{\mathrm{delay}}$, $T_{\mathrm{delay}} = T_h + T_p\left(1 - \mathtt{R}(T_{\mathrm{prop}}, T_p)\right)$, and $T_h = h/R$. T_{delay} represents the delay to send the ACK because it is piggybacked to the header of the coded packets. The function $\mathtt{R}(x,y)$ returns the remainder of x/y.

2) Round Robin Data Sharing in the Full-Duplex Channel (DSRR Full Duplex): We consider the simpler problem of $M_1 = M_2 = M$ and assume that both nodes start transmitting at the same time. The kth packet of each node is transmitted every $(mM+k)T_p$ for $m = 0,1,2,\ldots$ until the other node gets all M packets. The sharing process is completed when both nodes have received all information. Thus

$$E[T] = T_w + T_p M\left(\gamma + E[\max_{i,k} X_k^i]\right) \quad (27)$$

where $1 + X_k^i$ is the number of transmissions of packet k needed to reach node i from the other node, $\gamma \in (1/2, 1)$, $T_w = 2T_{\mathrm{prop}} + T_{\mathrm{delay}}$, $T_{\mathrm{delay}} = T_h + T_p\left(1 - \mathtt{R}(T_{\mathrm{prop}}, T_p)\right)$, $T_h = h/R$, and $E[\max_{i,k} X_k^i] = \sum_{t=1}^{\infty}\left[1 - (1 - Pe^t)^{2M}\right]$. An upper and lower bound on the mean completion time is given by $\gamma = 1$ and $\gamma = 1/2$, respectively.

3) Round Robin Data Sharing in TDD Channel: This approach is similar to DSRR Full Duplex but considers a TDD channel. Each transmitter sends all M packets before stopping to listen for a transmission of the other. The data packets also contain feedback indicating if the node should keep transmitting or if all packets have been received successfully at the other node. The mean completion time $E[T]$ is bounded using the same definitions as in DSRR Full Duplex as follows:

$$E[T] \geq (T_w + T_p M)\left(1 + E[\max_{i,k} X_k^i]\right) + T_p M \quad (28)$$

$$E[T] \leq (T_w + 2T_p M)\left(1 + E[\max_{i,k} X_k^i]\right). \quad (29)$$

We provide now numerical results assuming symmetric uplink and downlink channels (i.e., $Pe_1 = Pe_2 = Pe$, $R_1 = R_2 = R = 1.5$ Mb/s, and $M_1 = M_2 = M$ packets. We choose the $N_{(i,j,t)}$'s using the proposed search algorithm under different packet erasure probabilities Pe.

Fig. 5 shows the mean completion time for our RLNC TDD scheme, two full-duplex schemes, and a TDD scheme with no coding. Fig. 5 illustrates that choosing $N_{(i,j,t)}$'s using the proposed search algorithm provides very good performance in terms of mean completion time for a wide range of Pe. In general, our scheme outperforms the TDD with no coding, although this is more noticeable for large Pe. For low Pe, our TDD scheme is only 1 dB away from the performance of the

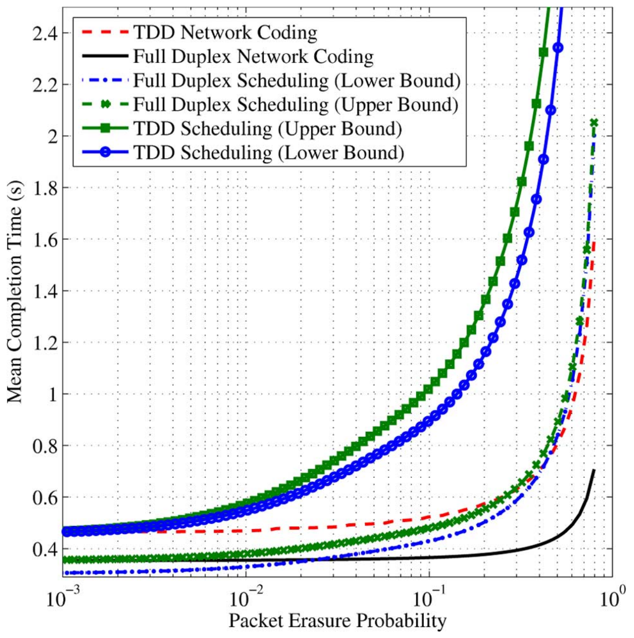

Fig. 5. Mean completion time for the TDD scheme choosing the $N_{(i,j,t)}$'s through the search algorithm proposed in Section V-B, two full-duplex schemes, and a TDD scheme with no coding. We use the following parameters: $R = 1.5$ Mb/s, $h = 80$ bits, $g = 20$ bits, and a block size per node of $M = 15$.

DSNC full-duplex scheme, which is optimal in mean completion time. Since we have packets to be transmitted from both nodes, we expected the difference between these two schemes to be around 3 dB. This result is explained because the T_{prop} is larger than the time for transmission of the coded packets. For very high Pe, this relationship is reversed and the gap is closer to the expected 3 dB.

Fig. 5 also shows that for $Pe \geq 0.4$, our TDD scheme clearly outperforms SDRR full duplex. However, it may do so as early as $Pe \geq 0.2$, considering the upper bound on SDRR full duplex. For $Pe = 0.8$, our TDD scheme outperforms SDRR full duplex by about 1.1 dB.

Finally, Fig. 5 shows that if we have a full-duplex system for two nodes to share data, they clearly should do so using network coding, especially for high Pe. Note that for $Pe = 0.8$, SDNC full duplex is more than 4 dB better in terms of the completion time performance than the scheduling scheme SDRR full duplex. We emphasize that for low Pe, the lower bound on SDRR full duplex is loose while the upper bound is tight. In fact, the performance of SDRR full duplex is always equal to or worse than that of SDNC full duplex.

VI. Practical Considerations

A. *Effect of Field Size*

Let us consider the effect of the field size from a receiver's perspective. This allows us to model the effect of the field size separately from other effects, such as, the channel or the network topology. We only assume that: 1) M data packets are combined using RLNC and that the network also uses RLNC at intermediate nodes and 2) the coded packets traverse a channel/network in which packets can suffer erasures. Thus, our model is useful to any RLNC network in which packet losses occur, regardless of the nature of the network (e.g., time-dependent losses, single or multihop network).

Using RLNC arguments, the process of decoding M packets at a receiver is modeled as a Markov chain . A transition occurs when a new coded packet is received, while the states in the Markov chain represent how many dofs are needed in order to decode all M data packets. The arrival of a new coded packet can cause either: 1) a transition to the next state, when it provides a new dofs or 2) a self-transition otherwise. The transition probability matrix for this problem is

$$P_q = \begin{bmatrix} q^{-M} & 1-q^{-M} & 0 & \cdots & 0 & 0 \\ 0 & q^{-M+1} & 1-q^{-M+1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & 0 & \cdots & q^{-1} & 1-q^{-1} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}.$$

Theorem 1 can be used to provide a full characterization of the number of coded packets that need to be received before successfully decoding the information.

Clearly, at least M-coded packets must be received before being able to decode. The following theorem provides an upper bound on the average number of coded packets that need to be received before decoding the M original packets.

Theorem 4: Let M be the number of data packets encoded using RLNC with a field size q, then the mean number of coded packets that have to be received before decoding the original packets is upper bounded by

$$\min\left\{M\frac{q}{q-1}, M+1+\frac{1-q^{-M+1}}{q-1}\right\}. \quad (30)$$

Proof: If N_c represents the minimum number of coded packets needed to decode, then $E[N_c] = \sum_{k=1}^{M} \frac{1}{1-q^{-k}}$. Since $q^{-k} \leq q^{-1}$ for $q \geq 2$ and $k \geq 1$, then $E[N_c] \leq \sum_{k=1}^{M} \frac{q}{q-1} = M\frac{q}{q-1}$ which shows the first bound. The second bound comes from $E[N_c] = M + \sum_{k=1}^{M} \frac{1}{q^k-1} \leq M + \sum_{k=0}^{M-1} q^{-k} = M + \frac{1-q^{-M}}{1-q^{-1}} = M+1+\frac{1-q^{-M+1}}{q-1}$ where we have used the fact that $q^k - 1 \geq q^{k-1}$ for $k \geq 1$ and $q \geq 2$. ∎

Remark: One important conclusion of Theorem 4 is that $E[N_c] \leq M+2, \forall\, q \geq 2$ (i.e., on average, the number of coded packets needed to decode the M original packets will be between M and $M+2$ for any field size). If $M \gg 2$, we expect that a scheme using $q = 2$ and one using larger q will have a small difference in their performance.

B. *Systematic Network Coding: Reducing the Decoding Complexity*

Systematic network coding consists of sending the original packets initially, and transmitting RLNC packets in subsequent transmissions. Essentially, systematic network coding introduces some structure to the code. However, it does cause a loss in performance from a delay or throughput perspective in a one-hop network, because the first transmissions of each original packet constitute new dofs to every receiver. Systematic network coding can also provide benefits from a decoding complexity perspective, because a fraction of the packets do not require any decoding [30]–[32]. However, previous work has not mathematically characterized the benefits on the decoding complexity.

We study the gains in average decoding complexity of systematic network coding as an alternative to pure RLNC. Although we focus our analysis on the case of an erasure channel in which packets can suffer erasures independently from other packets (IID Bernoulli with parameter Pe), the results apply to more general networks which can be translated into an equivalent erasure channel. Also, the techniques and mechanisms used to derive our results can be extended to characterize time-dependent erasure channels.

1) Model: A source node transmits the original packets followed by RLNC packets. RLNC packets are transmitted until the receivers have enough dofs to decode. We assume a packet erasure channel where erasures are IID Bernoulli with parameter Pe. We also assume that the decoder can recognize uncoded packets and use this knowledge to speed up the decoding process.

A receiver with $D \leq M$-uncoded packets can decode the remaining $M - D$-coded packets in $O((M-D)^3)$ operations if it uses Gaussian elimination. We show this by considering that the matrix of coefficients can be reordered as

$$[CP_j] = \begin{bmatrix} 1 & 0 & \cdot\cdot & 0 & \cdot\cdot & 0 \\ \vdots & \vdots & \vdots & \vdots & & \\ 0 & 0 & \cdot\cdot & 1 & \cdot\cdot & 0 \\ a_{(D+1)1} & a_{(D+1)2} & \cdot\cdot & a_{(D+1)D} & \cdot\cdot & a_{(D+1)M} \\ \vdots & \vdots & \vdots & \vdots & & \\ a_{M1} & a_{M2} & \cdot\cdot & a_{MD} & \cdot\cdot & a_{MM} \end{bmatrix} P'$$

where $[CP_j]$ constitutes the vector of (coded) packets received, and P' is the vector of original packets in the appropriate order to have the adequate matrix structure. Uncoded packets are used to perform a forward elimination only in the coded packets that are received and not in the other uncoded packets. Furthermore, the operation is restricted to a single column in the matrix corresponding to the equivalent uncoded packet. Finally, no backward substitution step is needed for the uncoded packets.

2) Decoding Complexity: Considering the characteristics of the channel, the following theorem characterizes the average decoding complexity of systematic network coding.

Theorem 5: The average number of operations required to decode using Gaussian elimination on a full-rank matrix when systematic network coding is used to transmit M original data packets, where each transmitted coded packet undergoes erasures that are IID Bernoulli with parameter Pe, grows as $O(M^3 Pe^3)$.

Proof: The number of uncoded packets D that are received depends only on the channel and the number M of original data packets that are transmitted, and it is given by a binomial distribution with probability Pe.

The number of operations for Gaussian elimination on a $r \times r$ matrix is $Ar^3 + Br^2 + Cr$, for some constants A, B, C. The number of operations required to decode the M original data packets is given by two effects: 1) elimination of contribution of uncoded packets in the linear combinations of coded packets, which requires $A_1(M-D)D$ operations, where A_1 is a constant, and 2) a full Gaussian elimination in the remaining $(M-D) \times (M-D)$ matrix, which requires $A(M-D)^3 + B(M-D)^2 + C(M-D)$ operations. Then, $E[A_1(M-D)D] = (A_1 Pe(1-Pe))(M^2 - M)$ and

$$E[A(M-D)^3 + B(M-D)^2 + C(M-D)] = A(MPe)^3 + (3A + B - 3APe)(MPe)^2 + \big(A - 3APe \quad (31)$$

$$+ 2APe^2 + B - BPe + C\big)(MPe). \quad (32)$$

The average number of operations is obtained by adding these two terms, which is $O(Pe^3 M^3)$. ■

This result shows that systematic network coding allows us to reduce computational complexity by a factor of Pe^3, on average, with respect to pure RLNC. Note that decoding RLNC-coded packets using Gaussian elimination on a full-rank matrix takes $O(M^3)$ operations.

Remark: In a channel with $Pe = 0.1$, on average, systematic network coding requires 1000 times fewer operations than RLNC in order to decode, for large enough M.

Let us consider the case in which Gaussian elimination is used not just to decode the information but also to determine if a newly received coded packet is useful (i.e., new dofs). The following theorem shows that the scaling $O(Pe^3 M^3)$ is preserved regardless of field size.

Theorem 6: The average number of operations required to decode, when systematic network coding is used and Gaussian elimination is performed on every new packet to determine linear independence, is upper bounded by a function that is $O(M^3 Pe^3)$, regardless of field size q, when the erasures are IID Bernoulli with parameter Pe.

Proof: We shall provide the proof for the case of multiplication operations (# Mult). We can use the Markov chain model that considers the effect of field size to solve this problem, but considering a different cost to the transitions. The number of operations required to process a newly received packet depends on whether the packet is uncoded (no operations) or not, how many uncoded packets were received (D), and how many linearly independent coded packets were previously received $(\beta - D)$. The number of multiplication operations to process a newly received packet are given by

$$D(1+K) + \mathbf{1}_{\{\beta > D\}} \sum_{u=1}^{\beta - D} (M - D - u + 1 + K) \quad (33)$$

where K represents the number of information symbols of size $log_2 q$ bits per packet and $\mathbf{1}_{\{s \in S\}}$ is 1 when $s \in S$ and zero otherwise. Let us determine the average number of operations conditioned on D-uncoded packets being received

$$E[\#\mathrm{Mult}|D] = \sum_{m=1}^{M-D} \frac{D+DK}{1-q^{-m}} + \kappa \sum_{m=1}^{M-D-1} \frac{1}{1-q^{-m}} + (-K-1/2) \sum_{m=1}^{M-D-1} \frac{m}{1-q^{-m}} + (-1/2) \sum_{m=1}^{M-D-1} \frac{m^2}{1-q^{-m}}$$

where $\kappa = \frac{-D}{2}(2M - D + 2K + 1) + M^2/2 + MK + M/2$, and the q^{-m} factors come from an RLNC argument. We have shown in Section VI-A that $\sum_{m=1}^{X} \frac{1}{1-q^{-m}} \leq X+2$ for any integer $X>0$ and $q \geq 2$. Using similar manipulations, we can show that $\sum_{m=1}^{M-D-1} \frac{m}{1-q^{-m}} \leq (M-D-1)(M-D)/2+4$. Finally, $\sum_{m=1}^{M-D-1} \frac{m^2}{1-q^{-m}} \geq \sum_{m=1}^{M-D-1} m^2$, which is a well-known series.

Let us compute $E[\#\text{Mult}] = \sum_d E[\#\text{Mult}|d]P(D=d)$, using the fact that $P(D=d)$ is characterized by a binomial distribution. After some manipulations, it can be shown that $E[\#\text{Mult}] = M^3Pe^3/3 + o(M^3)$. The proof concludes by noting that the operations related to the backward substitution step in Gaussian elimination grow as $O(M^2Pe^2)$ and are performed only once after enough dofs have been received, regardless of field size q. ■

Remark: For RLNC performing Gaussian elimination at a cost of AM^3+BM^2+CM operations, the term AM^3, becomes the dominant effect after a certain value of M. For packets with K data symbols, typical values are $A=1/3$, $B=(K+1)/2$, and $C=(3K-5)/6$. Thus, for $M^3/3$ to be dominant, we require $M \gg 3K/2+3/2$. As an example, if $K=1000$ symbols per packet, then $M \gg 1500$. Since $(K+1)M/2 > (3K-5)/6, \forall\, M \geq 1$, the term BM^2 is dominant for $M \ll 3K/2+3/2 \approx 1500$.

The average number of multiplications in systematic network coding in the same example is given by $A'(MPe)^3 + B'(MPe)^2 + C'(MPe)$ operations, where $A'=1/3$, $B'=((K+3)/2-Pe+Pe(1-Pe))$, and $C'=(1/3-Pe+2/3Pe^2+(K+1)/2-(K+1)/2Pe+(3K-5)/6-Pe(1-Pe))$. For our example, the cubic term is dominant if $M \gg (3K/2+9/2-Pe^2)/Pe \approx 1500/Pe$. Note that if we fix $Pe=0.01$, this means that the cost will grow essentially as M^2 up to $M<1500$. Many applications rely on coding within this region. Since the scaling in systematic network coding is actually dependent on the pair MPe rather than on M alone, this causes the cubic increase in complexity effect to be dominant at much larger values of M. Finally, for $MPe \ll C'/B'$, the average complexity of systematic network coding increases linearly.

C. Example of Systematic Network Coding in TDD Channels

We study the case of one source and one receiver to illustrate: 1) the model modifications to characterize systematic network coding and 2) the effect of field size. The general structure of the scheme and the ACK packets is preserved. The first transmission contains $N_s \geq M$ packets, with M original data packets followed by N_s-M RLNC packets. We incorporate this structure in the Markov chain model by including a "systematic" state (State S). This state is the first to be visited and is only visited once. The remaining states are the same as our TDD RLNC scheme.

We consider now the transitions from the systematic state S to all other states. There is no self-transition to S. The transition probability to go from state S to state i is given by

$$P_{S\to i} = (1-Pe_{\text{ack}}) \sum_{j=0}^{M-i} P\left(M-i|j \text{ uncoded}, S\right) \mathbf{P}_{(M,M,Pe)}(j)$$

for $i \neq M$. Finally, $P_{S\to M} = 1 - \sum_{j=0}^{M-1} P_{S\to j}$.

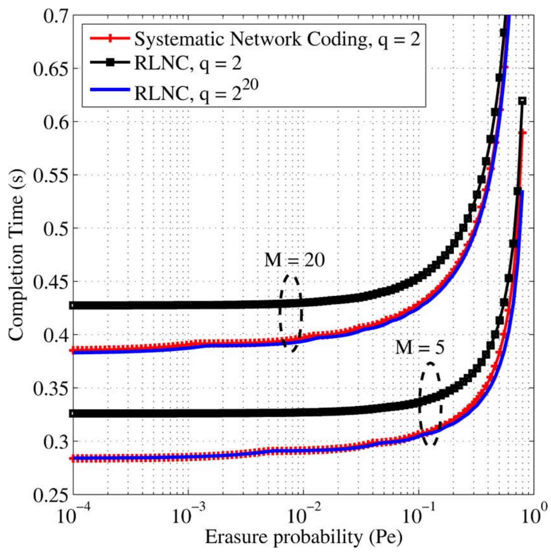

Fig. 6. Mean completion time for the TDD scheme with different M and field sizes $q=2$ and $q=1048576$. We use the following parameters $R=1.5$ Mb/s, $h=80$ bits, $n_{\text{ack}}=100$ bits, $n=10{,}000$ bits.

We define the probability that $M-i$ dofs are correctly received, given that the system is in state S and j uncoded packets have been received as

$$P\left(M-i|j \text{ uncoded}, S\right) = P\left(M-i-j \text{ coded}|S\right) = \sum_{l=M-i-j}^{N_s-M} P\left[M-i-j \text{ coded}|S, l \text{ received}\right] \mathbf{P}_{(M,N_s-M,Pe)}(l)$$

which is equivalent to the probability of $M-i-j$-coded packets being received. The term $P\left[M-i-j \text{ coded}|S, l \text{ received}\right]$ can be found by computing P_q^l, using the transition probability matrix P_q computed in Section VI-A, and searching in the appropriate column and row corresponding to starting state M and end state $M-i-j$.

For starting from states other than the systematic state S, we assume that $N_i \geq i$. For $i > i'$, we have

$$P_{i\to i'} = (1-Pe_{\text{ack}}) \sum_{k=\max\{1,i-i'\}}^{N_i} P\left[i'|i,k\right] \mathbf{P}_{(i,N_i,Pe)}(k)$$

where k represents the number of coded packets that have been received. $P\left[i'|i,k\right]$ represents the probability of starting at state i and transitioning to state i' in k transitions or hops. This can be found by computing P_q^k, and searching in the appropriate column and row corresponding to starting state i and end state i'. For the case of $i=i'>0$

$$P_{i\to i} = (1-Pe_{\text{ack}}) \left[\sum_{k=0}^{N_i} P\left[i|i,k\right] \mathbf{P}_{(i,N_i,Pe)}(k)\right] + Pe_{\text{ack}}$$

and that $P_{0\to 0}=1$.

D. Numerical Results

Fig. 6 shows the mean completion time for the RLNC TDD scheme for a single receiver for $q=2$ and $q=2^{20}$ for various block sizes M and Pe from 10^{-4} to 0.8. It illustrates that the gap between field sizes $q=2$ and $q=2^{20}$ is very small. The completion time is increased by, at most, 15% and 10% on average for $M=5$ and $M=20$, respectively. Clearly, the degradation in performance is small even for small M and q.

Fig. 6 shows that the performance of systematic network coding with $q = 2$ is essentially the same as RLNC with a large field size for moderate Pe, while their difference at high Pe is very small. RLNC with field size $q = 2$ constitutes an upper bound to the mean completion time of the systematic network coding scheme with the same q. Since increasing M closes the gap between RLNC with large q and with $q = 2$, then the gap between our systematic approach and RLNC with high q also closes.

VII. Operation Under Random Arrivals

The assumption up to this point is that the source had M data packets in its buffer before starting transmission. We now consider that this buffer may sometimes be empty or contain fewer than M packets awaiting transmission. Then, the source node must choose to either wait for additional packets to arrive, or take those packets in the buffer and start performing RLNC.

We first study an online network coding scheme for TDD channels under Poisson arrivals. This problem is modeled as a bulk service queue with a general service process that depends on the batch size, but where packets that are being serviced can be fed back into the queue. Second, we present the necessary changes to the queuing model derived for online network coding in order to analyze our RLNC scheme for TDD channels described in Section IV. This problem requires a similar queue model but feeding back packets into the queue.

A. Online Approach

1) Preliminaries: We can think of packets as vectors over a finite field. Since we focus on linear network coding, we can think of the state of knowledge of a node as a vector space over the field. Reference [26] showed that with the proper use of feedback, it is possible to perform network coding in an online manner. The authors relied on acknowledging every new dof that was successfully delivered to a receiver. Reference [26] showed that using the feedback on dofs required the queue to store a basis for a coset space with respect to the subspace of knowledge common to all of the receivers. The authors defined a specific way of computing this basis using the notion of a node "seeing" a data packet, which we define in the following text.

Definition: Index of a packet: For any integer $k > 0$, the kth packet that arrives at the sender is said to have index k.

Definition: Seeing a packet: A node is said to have "seen" an original packet p_k, with index k, if it has received enough information to compute a linear combination of the form $(p_k + q)$, where q is itself a linear combination involving only packets with an index greater than that of p_k (i.e., of an index greater than k). Note that decoding a packet implies seeing that packet, which corresponds to $q = 0$.

Our feedback scheme is inspired by [26]. However, the TDD constraint requires us to use feedback more sporadically (i.e., we cannot send an ACK for every received dofs). Instead, we determine the number of coded packets to transmit before the sender stops to listen for an ACK. This ACK will report the last consecutive "seen" packet, say of index k, to allow the sender to remove all packets with index k or less.

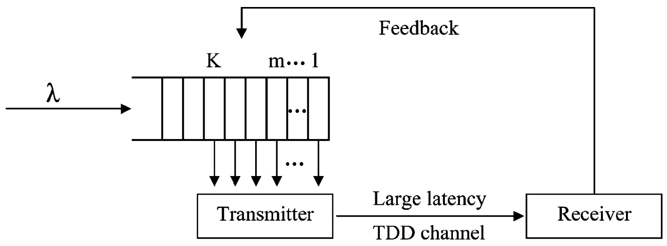

Fig. 7. Bulk queue model with feedback for online network coding for TDD channels.

We do not guarantee that the seen packets will be decoded immediately, similar to the decode-when-seen algorithm in [26]. In general, there is a delay in decoding the data packets, because the receiver has to collect enough dofs involving the unknown packets. We shall characterize how often a decoding event occurs, which will be defined.

Definition: Decoding event: For any positive integer k, if random combinations at the sender have involved up to the kth packet, it is said that a decoding event occurs if the receiver decodes the kth packet and all packets before it after a transmission of a group of coded packets by the sender. In other words, it constitutes the decoding at the receiver of all packets that were involved in random linear combinations up to that moment.

2) System Model: We consider the case of one sender and one receiver, with similar assumptions as in previous sections. The key difference is that the packets included in a linear combination at a given time are determined by the ACK packets received up to that time, the packets in the queue, and the window of coding K. We consider that the data packets arrive to a source node through a Poisson process with rate λ packets/s. Upon arrival, the data packet is placed in a buffer to await encoding and transmission to the receiver, as in Fig. 7. The buffer forms a first-in-first-out (FIFO) queue. However, some of the packets will be fed back to the queue because they were not successfully delivered. That is, the ACK packet from the receiver contains an index number that is lower than the index of some of the packets used in the generation of the previous batch of coded packets.

The size of the coding window of packets is variable, where $m \le M \le K$. The pair (m, K) constitutes the range of the bulk size or the size of the coding window used to perform RLNC. If the buffer has fewer than m data packets, the system will wait until m packets arrive before providing service. If the buffer contains more than K packets, the system will service exactly K packets. Finally, if the buffer has M packets with $m \le M \le K$, then the system will service M packets.

Since we are potentially mixing a different number of data packets, we assume that the coded packet contains space for the maximum number of coefficients allowed by the window (i.e., K for simplicity). Therefore, the number of bits in each coded packet is $h + n + gK$. The ACK packet feeds back the index of the last consecutive seen data packet also indicating the number of dofs that are still required to decode. The number of coded packets sent back-to-back depends on the value of M (i.e., the number of packets that will be included in the linear combinations). The service time depends on the number of data packets taken from the queue at any time (i.e., the service time distribution is general but it depends on the size of the batch being transmitted).

Transmission begins after an ACK packet is received and $j \geq m$ packets are in the queue. At this point, j information packets are taken from the queue, which are encoded into $N_j \geq j$-coded packets, and transmitted. The ACK informs the transmitter about the index of the last consecutively seen packet. At this point, the source may have received new data packets. If i packets will be used to generate linear combinations in the next round of transmission, then the transmitter sends N_i-coded packets. If the number of packets in the queue exceeds K, then only $i = K$ packets are involved in the linear combinations of the next transmission. The time between a transmission of packets and receiving an ACK packet is $t_i = N_i T_p + T_w$ for $i = m, \ldots, K$. For $i > K$, the transmission time is t_K.

3) Queueing Model: This bulk queueing model considers Poisson arrivals and a general service time that depends on the bulk size. We consider a minimum batch size m that could be different from 1 (which was the assumption in [33]). Also, the system model is very similar to the bulk queue model studied in [34] but considering a nonzero probability of packets being fed back to the queue. Thus, our work generalizes [33] and [34]. The transition probability of the number of packets in the queue is given by

$$\wp = \begin{bmatrix} a_0^{(m)} & a_1^{(m)} & \cdots & a_K^{(m)} & a_{K+1}^{(m)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(m)} & a_1^{(m)} & \cdots & a_K^{(m)} & a_{K+1}^{(m)} & \cdots \\ a_0^{(m+1)} & a_1^{(m+1)} & \cdots & a_K^{(m+1)} & a_{K+1}^{(m+1)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_0^{(K)} & a_1^{(K)} & \cdots & a_K^{(K)} & a_{K+1}^{(K)} & \cdots \\ 0 & a_0^{(K)} & \cdots & a_{K-1}^{(K)} & a_K^{(K)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

where $a_k^{(j)}$ is the probability of having k arrivals plus fed-back packets during a service of type j.

4) MGF of Transition Probabilities: The MGF of the transition probabilities is determined by considering that the new state of the queue (q_{new}) depends on the previous state of the queue (q_{old})), the Poisson packet arrivals $(A(q_{old}))$, and packet departures (taken out of the queue) given the ACK information $(D(q_{\text{old}}))$. More specifically, $q_{\text{new}} = q_{\text{old}} + A(q_{\text{old}}) - D(q_{\text{old}})$. The distribution of packet departures given a previous state i, a set of values of N_i's given by $\{N_i\}$, and an erasure probability Pe is given by

$$\mathcal{P}_{(i,\{N_i\},Pe)}(k) = \begin{cases} \mathbf{P}_{(i,N_i,Pe)}(k), & \text{if } i \leq K \\ \mathbf{P}_{(K,N_K,Pe)}(k), & \text{if } i > K \end{cases}$$

while their MGF is

$$M_{(i,N_i,Pe)}^{(i)}(s) = \textstyle\sum_{k=0}^{k=i} e^{sk} \mathcal{P}_{(i,N_i,Pe)}(k) = \quad (34)$$

$$\textstyle\sum_{k=0}^{i-1} \left(\binom{N_i}{k} \left(\frac{1-Pe}{Pe}\right)^k Pe^{N_i} \left[e^{sk} - e^{si}\right] \right) + e^{si}. \quad (35)$$

Using basic MGF properties, the MGF of the transition probabilities when the system is in state i is given by

$$M_T^{(i)}(s) = e^{\lambda t_i}(e^s - 1) e^{si} M_{(i,N_i,Pe)}^{(i)}(-s). \quad (36)$$

5) Stationary Probabilities: We define the z-transform of the transition probabilities as

$$T^{(j)}(z) = \textstyle\sum_{k=0}^{\infty} a_k^{(j)} z^k =$$
$$e^{\lambda t_i}(z-1) \left[\textstyle\sum_{k=0}^{i-1} \left(\binom{N_i}{k} \left(\frac{1-Pe}{Pe}\right)^k Pe^{N_i} \left[z^{i-k} - 1\right] \right) + 1 \right]$$

where $a_k^{(j)} = \frac{1}{k!} \frac{\partial^k}{\partial z^k} T^{(j)}(z) \Big|_{z=0}$.

We define $\Pi(z) = \sum_{i=0}^{\infty} \pi_i z^i$ as the corresponding generating function of the stationary probabilities, where π_i is the stationary probability of state i. Reference [34] showed that $\Pi(z)$ can be expressed as

$$\Pi(z) = \frac{T^{(K)}(z) \sum_{i=0}^{K} \pi_i z^i - z^K T^{(m)}(z) \sum_{i=0}^{m} \pi_i}{T^{(K)}(z) - z^K} - \frac{\sum_{i=m+1}^{K} \pi_i T^{(i)}(z)}{T^{(K)}(z) - z^K}$$

which provides an expression for $\Pi(z)$ in terms of its first $K+1$ coefficients $\pi_0, \ldots, \pi_K$. Determining these $K+1$ coefficients provides a full characterization of the stationary probabilities [34].

Using the same techniques as [34], we can prove that $T^{(K)}(z) - z^K$ has exactly K zeros satisfying $|z| \leq 1$ assuming that $T^{(K)}(z)$ has a radius of convergence greater than one. Denoting the roots as $1, z_1, \ldots, z_{K-1}$ and assuming that they are different, then the numerator of (37) has to vanish for $z_1, \ldots, z_{K-1}$ which gives us $K-1$ linear equations

$$T^{(K)}(z_k) \sum_{i=0}^{K} \pi_i z_k^i - z_k^K T^{(m)}(z_k) \sum_{i=0}^{m} \pi_i \quad (37)$$
$$- \sum_{i=m+1}^{K} \pi_i T^{(i)}(z_k) = 0 \quad (38)$$

for $k = 1, \ldots, K-1$. Also, the numerator vanishes trivially for $z = 1$ for the numerator and the denominator in (37). We use l'Hôspital's rule to exploit the fact that $\Pi(1) = 1$ to derive an additional linear equation given by

$$1 = \sum_{i=0}^{m} \left[1 + \frac{i - \theta_m}{\theta_K - K} \right] \pi_i + \sum_{i=m+1}^{K} \left[\frac{\theta_K + i - \theta_i}{\theta_K - K} \right] \pi_i \quad (39)$$

where we have defined $\frac{\partial T^{(i)}(z)}{\partial z} \Big|_{z=1} = \theta_i$. The final linear equation to fully characterize $\Pi(z)$ is given by

$$(a_0^m - 1)\pi_0 + a_0^m \pi_1 + \cdots + a_0^m \pi_m + a_0^{m+1} \pi_{m+1} + a_0^K \pi_K = 0.$$

We can use techniques developed by [35] to compute the roots of $T^{(K)}(z) - z^K$. In fact, a simple solver is sufficient to find the root of $z\left(T^{(K)}(z)\right)^{-1/K} - e^{2ki\pi/K}$ with $i = \sqrt{-1}$, for every value of $k \in \{0, \ldots, K-1\}$, which provides us with the K required roots.

6) Queue of Finite Capacity: The general solution requires the calculation of the roots of $T^{(K)}(z) - z^K$, which can result in numerical inaccuracies and becomes increasingly difficult when the decision variable K assumes a larger value. For these reasons, we simplify the problem considering that the system has a capacity of B packets waiting to be serviced. The transition probability matrix for this case has a similar structure as $\wp$ but with 1) finite dimensions $((B+1) \times (B+1))$, and 2) a final column that aggregates the probability of transitioning to states larger than B.

The stationary distribution is computed by solving $\bar{\pi} = \wp\bar{\pi}$, with $\bar{\pi} = [\pi_0, \ldots, \pi_B]^T$, under the constraint that $\sum_{i=0}^{B} \pi_i = 1$.

7) Mean Delay: We define the mean delay $E[D]$ of a packet as the time that elapses since a packet arrives to the queue to the time it is "seen" at the receiver. For this purpose, let us first define the mean queue size as $E[Q] = \sum_{i=0}^{B} i\pi_i$ for the case in which the queue capacity is B. Let us also define $T_{(m,K)}$ as the transmission time for a choice of (m, K). Then

$$E[T_{(m,K)}] = \left[t_m \sum_{i=0}^{m} \pi_i + \sum_{i=m+1}^{K-1} t_i \pi_i + t_K \sum_{i=K}^{B} \pi_i \right]$$

where $\sum_{i=K}^{B} \pi_i = 1 - \sum_{i=0}^{K-1} \pi_i$. Using Little's law

$$E[D] = E[Q]/\lambda + E[T_{(m,K)}]. \tag{40}$$

Remark: In our case, $t_i, \forall\ i$ is a constant value. If it were random, the expression for $E[T_{(m,K)}]$ would remain the same, but with t_i representing the mean time for a service of type i.

8) Mean Time Between Decoding Events: Since data packets can be seen without being decoded, we now characterize the time between decoding events, which provides a worst case metric for the time between a packet is seen until it is decoded. We analyze this for the case of $m = 1$ and for a finite capacity queue, but an extension to general m and infinite B follow naturally.

The time between decoding events (D_e) is calculated by modeling the problem as an absorbing Markov chain. The absorption state (State 0) indicates that a decoding event occurred. Other states correspond to the number of packets in the queue. A transition to the absorbing state may take place from any other state. We define $P_a(j|i) = \frac{e^{-\lambda t_i}(\lambda t_i)^j}{j!}$, $Rd(k,l) = 1 - \sum_{j=0}^{k} d_j^{(l)}$, and $d_j^{(i)} = \sum_{k=\max\{0,j-i\}}^{j-1} P_a(k|i)\mathcal{P}_{(i,\{N_i\},Pe)}(i + k - j)$ for $i \in \{1, \ldots, K\}$. Finally, we define $d_0^{(i)} = \mathcal{P}_{(i,\{N_i\},Pe)}(i)$. The transition probability matrix $\mathcal{P}_A$ is of the form

$$\mathcal{P}_A = \begin{bmatrix} 1 & 0 & & \cdots & 0 & 0 \\ d_0^{(1)} & d_1^{(1)} & & \cdots & d_{B-1}^{(1)} & Rd(B-1,1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ d_0^{(K)} & d_1^{(K)} & & \cdots & d_{B-1}^{(K)} & Rd(B-1,K) \\ d_0^{(K)} & 0 & d_1^{(K)} & \cdots & d_{B-2}^{(K)} & Rd(B-2,K) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ d_0^{(K)} & \cdots & d_1^{(K)} & \cdots & d_{B-K}^{(K)} & Rd(B-K,K) \end{bmatrix}.$$

The time between decoding events depends on the starting state. The probability of starting at state i in this absorbing Markov chain depends on the state of the queue and the fact that the previous transmission resulted in packets being seen but not decoded. We determine the probability of starting in state i by using the stationary probabilities of Section VII-A-VI. Each state i in Section VII-A-VI, inherently contains two states: (i, d) and (i, n), where d and n indicate if the last transmission caused a decoding event, respectively. We define $\pi_{j,d}$ and $\pi_{j,n}$ as the stationary probabilities associated with the inherent states (i, d) and (i, n), where $\pi_j = \pi_{j,d} + \pi_{j,n}$. Let $P_{i,f\to j,f'}$ be the transition probability from state (i, f) to state (j, f'), with $f, f' \in \{d, n\}$. We also define $P_{i\to j,*} = P_{i,d\to j,*}\frac{\pi_{i,d}}{\pi_i} + P_{i,s\to j,*}\frac{\pi_{i,s}}{\pi_i}$ for $* \in \{d, n\}$ as the transition probability from state i (or rather from both (i, d) and (i, n)) to state $(j, *)$.

The mean time between decoding events is

$$E[D_e] = \sum_{i\geq 1} E[D_e|i]\pi_{i,n} \tag{41}$$

where $E[D_e|i]$ represents the mean time to absorption in the Markov chain given that the system started in state i, where the transition time is t_i. Note that for $* \in d, n$ $\pi_{j,*} = \sum_{i,s\in\{d,n\}} P_{i,s\to j,*}\pi_{i,s} = \sum_i P_{i\to j,*}\pi_i$. We are interested in determining $\pi_{j,n}$. For simplicity, we determine $\pi_{j,d}$ through $P_{i\to j,d}$, which is given by

$$P_{i\to j,d} = \begin{cases} \mathcal{P}_{(i,\{N_i\},Pe)}(i)P_a(j|i), & \text{if } i \leq K \\ \mathcal{P}_{(i,\{N_i\},Pe)}(K)P_a(j-(i-K)|K), & \text{if } i > K \end{cases}$$

and then determine $\pi_{j,n} = \pi_j - \pi_{j,d}$. After some manipulations, we show that for $j \geq 1$

$$\begin{aligned} \pi_{j,d} = \frac{\lambda^j}{j!}\sum\nolimits_{i=1}^{K} \left[\pi_i e^{-\lambda t_i} t_i^j \mathcal{P}_{(i,\{N_i\},Pe)}(i)\right] \\ + \sum\nolimits_{i=K+1}^{B} \frac{\pi_i e^{-\lambda t_K}(\lambda t_K)^{j-(i-K)}}{(j-(i-K))!}\left[\mathcal{P}_{(i,\{N_i\},Pe)}(K)\right] \\ + \frac{e^{-\lambda t_1}(\lambda t_1)^j}{j!}\pi_0\left[\mathcal{P}_{(1,\{N_i\},Pe)}(1)\right]. \end{aligned} \tag{42}$$

B. Batch-by-Batch Approach

Only small changes are required to analyze the queuing performance of our batch-by-batch TDD RLNC scheme. The service time of the queue is given by the time it takes to transmit a group of M packets taken from the queue using RLNC for TDD channels, as studied in previous sections, but with no packets being fed back to the queue. The service time is general and it depends on the number of data packets taken from the queue at any time. This constitutes a bulk queueing model $M/G^{(m,K)}/1$ [34] and is studied in detail in [36]. Changes are necessary from our discussion of the online mode as follows.

- The parameter $a_k^{(j)}$ represents the probability of k arrivals during a service of type j.
- The z-transform of the stationary probabilities $T^{(j)}(z) = \sum_{k=0}^{\infty} a_k^{(j)} z^k = M_{T,j}(\lambda(z-1))$ and $M_{T,j}(s)$ is the MGF of the completion time for the case of one sender and one receiver.
- The parameter can be easily computed as $a_k^{(j)} = \frac{1}{k!}\frac{\partial^k}{\partial z^k}M_{T,j}(\lambda(z-1))\Big|_{z=0}$.
- The system is stable if and only if $\lambda < K\mu_K$, where $1/\mu_K$ is the mean service time when the bulk size is $M = K$, where $1/\mu_j = \frac{\partial}{\partial z}M_{T,j}(z)\Big|_{z=0}$.
- In this case, $\frac{\partial T^{(i)}(z)}{\partial z}\Big|_{z=1} = \theta_i = \lambda/\mu_i$ since $\lambda/\mu_K \neq K$ in general, and the denominator of (39) will not become zero as $z \to 1$. In fact, if the system is stable, this condition will be satisfied.

C. *Performance Analysis and Numerical Results*

We present different policies to choose the values of N_i's in our online approach to illustrate the performance of the system and its relationship to the choice of N_i's.

a) Optimizing for Mean Delay: Computes N_i's following an exhaustive search method in order to minimize $E[D]$.

b) Optimizing for Mean Time between Decoding Events: Computes N_i's following an exhaustive search method in order to minimize $E[D_e]$.

c) Heuristics 1: This heuristics computes $N_i = \lfloor \frac{i}{1-Pe} \rfloor$.

d) Heuristics 2: This heuristics computes $N_i = \lceil \frac{i}{1-Pe} \rceil$.

e) Heuristics 3: This heuristics computes N_i's so that a minimum criteria for the probability of generating a decoding event in the next transmission is achieved. We specify ϵ as the minimum acceptable probability of decoding in the next transmission and compute N_i's $\forall\ i \in m, \ldots, K$ so that $\mathcal{P}_{(i,\{N_i\},Pe)}(i) \geq \epsilon$.

Fig. 8(a) and (b) shows the performance of the different schemes in terms of mean delay and mean time between decoding events, respectively, when the arrival rate has changed. A good mean delay performance does not necessarily imply a poor mean time between decoding events performance, or vice-versa. For example, Heuristics 1 for small λ shows that it is possible to have poor performance in both metrics. Fig. 8(a) and (b) also illustrates that the optimal choice of N_i's also depends on arrival rate λ for both metrics. However, Heuristics 2 shows a good tradeoff between mean delay and mean time between decoding events, while choosing N_i's independently from λ. For Heuristics 2, the mean time between decoding events is an order of magnitude smaller than its mean delay performance (i.e., $E[D]$ is the main contributor to the overall delay of a packet) from being received to being decoded.

Fig. 8(a) also illustrates that all schemes change smoothly with λ in terms of mean delay, except the optimizing for mean time between decoding events scheme, which shows a jagged curve. The main reason for this behavior is that the optimal values of N_i's in terms of $E[D_e]$ change considerably with respect to λ. For small λ, there might be numerical inaccuracies in computing the values of N_i's that minimize $E[D_e]$, due to the very small values reported in Fig. 8(b). These effects translate into a smooth change in the $E[D_e]$ [Fig. 8(b)] but with nonsmooth behavior in mean delay.

Finally, choosing the N_i's to increase the probability of decoding in the next transmission, as in Heuristics 3, shows poor performance in both metrics and depends on ϵ.

The optimal values of N_i depend not only on system and channel characteristics, for example, Pe, T_{prop}, T_p, but also on λ.

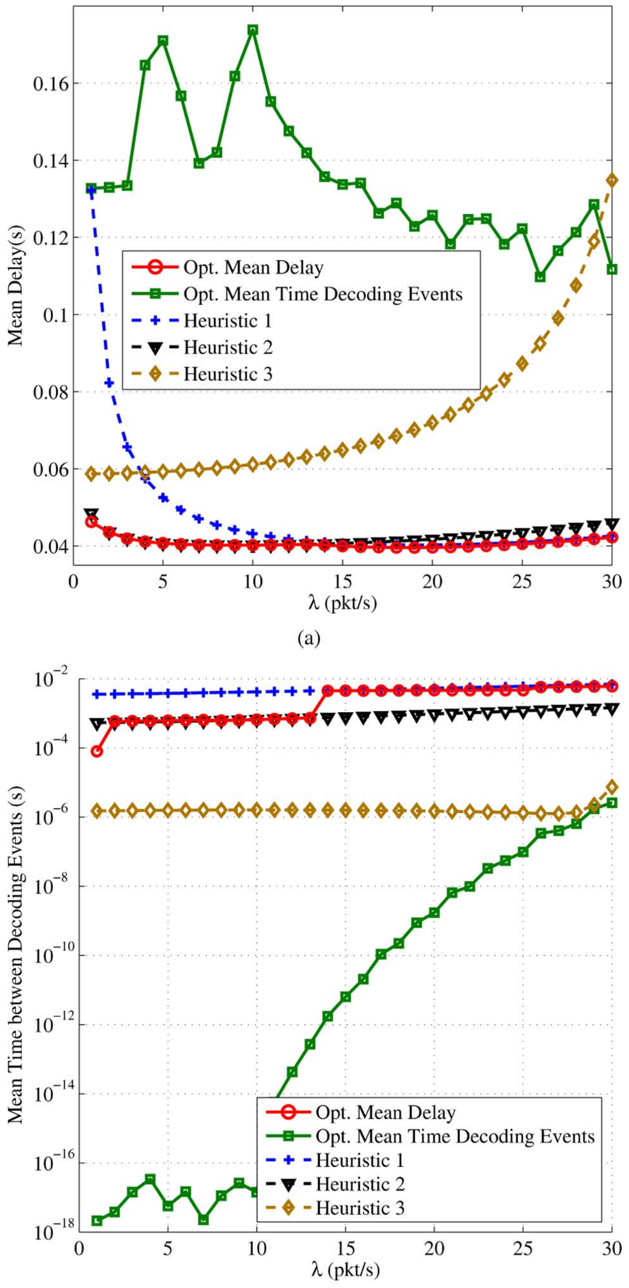

Fig. 8. Mean delay (a) and mean time between decoding events (b) of different schemes for choosing N_i's for the case of $(m, K) = (1, 3)$. Parameters $Pe = 0.1$, propagation time of 12.5 ms, data packets of 10 000 bits, $g = 20$ bits, $R = 1.5$ Mb/s, $h = 80$ bits, $B = 30$ packets, and the ACK packet has 100 bits. For Heuristics 3, $\epsilon = 0.99999$.

VIII. Conclusions

We considered the role of network coding with feedback in TDD channels and proposed a new paradigm for delay-driven coding, which is key to the design of efficient systems with large latency. By providing full characterization of simple and adaptive RLNC schemes for one-to-all broadcast and all-to-all broadcast, we were able to identify key parameters and develop estimation techniques to tailor coding and feedback. One of the most striking results is that our TDD schemes achieved the same or close to the same performance as the delay-optimal full-duplex scheme. Another key observation is that RLNC needs on average, at most, $M+2$-coded packets at any receiver to decode the original M packets, regardless of the field size. Clearly, the overhead is negligible for large M.

Perhaps the most important practical implication is that systematic RLNC using $\mathbf{GF}(2)$ provides a free lunch by: 1) providing close to optimal performance; 2) relying on simple operations, that is, XORs, for encoding/decoding; and 3) reducing the decoding complexity by a factor of Pe^3 with respect to using an RLNC with the same field size.

Our analysis was given for the case of one-hop communication, but the procedures and insights can be translated to more complex networks, even for systematic RLNC. For the latter, a key problem is to identify the equivalent Pe for each receiver. For example, a daisy chain with an erasure probability of Pe_i for link i will have an equivalent $Pe = 1 - \prod_i (1 - Pe_i)$. Future research will extend systematic RLNC efficiently to more complex networks. The main challenge is to determine which nodes and/or links should preserve the systematic structure.

Coding for delay is a topic that warrants further study. This work has identified important facets, for example, latency, limited feedback, and TDD channels, but many open problems still remain. One interesting example is to consider packets with deadlines, as in real-time applications.

Finally, studying the impact of half-duplex constraints on existing theoretical results and on practical schemes is a relevant aspect for future research on wireless networks. Although the conclusion for some of our work is that a half-duplex system could reach close to or have the same performance as a system operating in full duplex, we emphasize that it required a channel-adaptive coding scheme to achieve this. Any given scheme is not guaranteed to have this trait.

References

[1] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[2] S. Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–371, Feb. 2003.

[3] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[4] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[5] D. S. Lun, M. Médard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *Phys. Commun.*, vol. 1, no. 1, pp. 3–20, Mar. 2008.

[6] D. S. Lun, P. Pakzad, C. Fragouli, M. Médard, and R. Koetter, "An analysis of finite-memory random linear coding on packet streams," in *Proc. 4th Int. Symp. Mod. Opt. Mob., Ad Hoc Wireless Netw.*, Boston, MA, Apr. 2006, pp. 1–6.

[7] P. Maymounkov, N. J. A. Harvey, and D. S. Lun, "Methods for efficient network coding," in *Proc. 44th Annu. Allerton Conf. Commun., Cont., Comput.*, Monticello, IL, Sep. 2006, pp. 482–491.

[8] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proc. SIGCOMM*, Kyoto, Japan, Aug. 2007, pp. 169–180.

[9] A. Eryilmaz, A. Ozdaglar, and M. Médard, "On delay performance gains from network coding," in *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, Princeton, NJ, Mar. 2006, pp. 864–870.

[10] E. Ahmed, A. Eryilmaz, M. Médard, and A. Ozdaglar, "On the scaling law of network coding gains in wireless networks," presented at the IEEE Mil. Commun. Conf., Orlando, FL, Oct. 2007.

[11] M. Ghaderi, D. Towsley, and J. Kurose, "Network coding performance for reliable multicast," in *Proc. IEEE Mil. Commun.. Conf.*, Orlando, FL, Oct. 2007, pp. 1–7.

[12] J. K. Sundararajan, D. Shah, and M. Médard, "ARQ for network coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Toronto, ON, Canada, Jul. 2008, pp. 1651–1655.

[13] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, "Capacity of wireless erasure networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 789–804, Mar. 2006.

[14] B. Shrader and A. Ephremides, "A queueing model for random linear coding," in *Proc. IEEE Mil. Commun. Conf.*, Orlando, FL, Oct. 2007, pp. 1–7.

[15] B. Shrader and A. Ephremides, "On the queueing delay of a multicast erasure channel," in *Proc. IEEE Inf. Theory Workshop*, Punta del Este, Uruguay, Oct. 2006, pp. 423–427.

[16] T. Ozugur, M. Naghshineh, P. Kermani, and J. A. Copeland, "On the performance of ARQ protocols in infrared networks," *Int. J. Commun. Syst.*, vol. 13, pp. 617–638, 2000.

[17] A. M. Shah, S. S. Ara, and M. Matsumoto, "An improved selective-repeat ARQ scheme for IrDA links at high bit error rate," in *Proc. 4th Int. Conf. Mob. Ubi. Mult.*, Christchurch, New Zealand, Dec. 2005, pp. 37–42.

[18] M. Stojanovic, "Optimization of a data link protocol for an underwater acoustic channel," in *Proc. Oceans—Eur.*, Brest, France, Jun. 2005, pp. 68–73.

[19] A. R. K. Sastry, "Improving automatic repeat-request (ARQ) performance on satellite channels under high error rate conditions," *IEEE Trans. Commun.*, vol. COM-23, no. 4, pp. 436–439, Apr. 1975.

[20] P. Yu and S. Lin, "An efficient selective-repeat ARQ scheme for satellite channels and its throughput analysis," *IEEE Trans. Commun.*, vol. COM-29, no. 3, pp. 353–363, Mar. 1981.

[21] I. F. Akyildiz, O. B. Akan, and J. Fang, "TCP-planet: A reliable transport protocol for interplanetary Internet," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 2, pp. 348–361, Feb. 2004.

[22] A. L. Duc, "Performance closed-form derivations and analysis of hybrid ARQ retransmission schemes in a cross-layer context," Ph.D. dissertation, Dept. Commun. Electron., Telecom ParisTech, Paris, France, 2010.

[23] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-repeat-request error control schemes," *IEEE Commun. Mag.*, vol. 22, no. 12, pp. 5–17, Dec. 1984.

[24] M. Luby, "LT-codes," in *Proc. 43rd Annu. IEEE FOCS*, Vancouver, BC, Canada, Nov. 2002, pp. 271–280.

[25] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[26] J. K. Sundararajan, D. Shah, and M. Médard, "On queueing for coded networks—Queue size follows degrees of freedom," in *Proc. IEEE Inf. Theory Workshop*, Bergen, Norway, Jul. 2007, pp. 1–6.

[27] M. T. Chao and W. E. Strawderman, "Negative moments of positive random variables," *J. Amer. Statis. Assoc.*, vol. 67, no. 338, pp. 429–431, Jun. 1972.

[28] N. Cressie, A. S. Davis, J. L. Folks, and G. E. Policello, II, "The moment-generating function and negative integer moments," *Amer. Stat.*, vol. 35, no. 3, pp. 148–150, Aug. 1981.

[29] D. E. Lucani, M. Médard, and M. Stojanovic, "Broadcasting in time-division duplexing: A random linear network coding approach," in *Proc. Works. Net. Cod., Theory App.*, Lausanne, Switzerland, Jun. 2009, pp. 62–67.

[30] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and T. Larsen, "Network coding for mobile devices—Systematic binary random rateless codes," in *Proc. Works. Cooper. Mob. Netw.*, Dresden, Germany, Jun. 2009, pp. 1–6.

[31] A. A. Yazdi, S. Sorour, S. Valaee, and R. Y. Kim, "Optimum network coding for delay sensitive applications in WiMAX unicast," in *Proc. IEEE Int. Conf. Com. Commun.*, Rio de Janeiro, Brazil, Apr. 2009, pp. 2576–2580.

[32] J. Barros, R. A. Costa, D. Munaretto, and J. Widmer, "Effective delay control in online network coding," in *Proc. IEEE Int. Conf. Com. Commun.*, Rio de Janeiro, Brazil, Apr. 2009, pp. 208–216.

[33] L. E. N. Delbrouck, "A feedback queueing system with batch arrivals, bulk service, and queue-dependent service time," *J. Assoc. Comput. Mach.*, vol. 17, no. 2, pp. 314–323, Apr. 1970.

[34] S. K. Bar-Lev, M. Parlar, D. Perry, W. Stadje, and F. A. Van der Duyn Schouten, "Applications of bulk queues to group testing models with incomplete identification," *Eur. J. Oper. Res.*, no. 183, pp. 226–237, 2007.

[35] W. B. Powell, "Stochastic delays in transportation terminals: New results in the theory and application of bulk queues," Ph.D. dissertation, Dept. Civil Eng., Mass. Inst. Technol., Cambridge, MA, 1981.

[36] D. E. Lucani, M. Médard, and M. Stojanovic, "Random linear network coding for time division duplexing: Queueing analysis," in *Proc. IEEE Int. Symp. Inf. Theory*, Seoul, Korea, Jun. 2009, pp. 1423–1427.

[37] D. E. Lucani, M. Stojanovic, and M. Médard, "Random linear network coding for time division duplexing: When to stop talking and start listening," in *Proc. IEEE Int. Conf. Com. Commun.*, Rio de Janeiro, Brazil, Apr. 2009, pp. 1800–1808.

[38] D. E. Lucani, M. Médard, and M. Stojanovic, "Random linear network coding for time division duplexing: Field size considerations," in *Proc. IEEE Glob. Commun. Conf.*, Honolulu, HI, Dec. 2009, pp. 1–6.

[39] D. E. Lucani, M. Médard, M. Stojanovic, and D. R. Karger, "Sharing information in time-division duplexing channels: A network coding approach," in *Proc. 47th Annu. Allerton Conf. Commun., Cont., and Comput.*, Monticello, IL, Sep. 2009, pp. 1403–1410.

[40] D. E. Lucani, M. Stojanovic, and M. Médard, "Random linear network coding for time division duplexing: Energy analysis," in *Proc. IEEE Int. Conf. Commun.*, Dresden, Germany, Jun. 2009, pp. 1–5.

Daniel E. Lucani (M'10) is an Assistant Professor at the Faculty of Engineering of the University of Porto and a member of the Instituto de Telecomunicações (IT). He received the B.S. (summa cum laude) and M.S. (with honors) degrees in Electronics Engineering from Universidad Simón Bolívar, Venezuela in 2005 and 2006, respectively, and the Ph.D. degree in Electrical Engineering from the Massachusetts Institute of Technology (MIT) in 2010. His research interests lie in the general areas of communications and networks, network coding, information theory and their applications to highly volatile wireless sensor networks, satellite and underwater networks, focusing on issues of robustness, reliability, delay, energy, and resource allocation. Prof. Lucani was a visiting professor at MIT. He is the general co-chair of the Network Coding Applications and Protocols Workshop (NC-Pro 2011) and has also served as reviewer for high impact international journals and conferences, such as, the IEEE JOURNAL OF SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON INFORMATION THEORY, IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE JOURNAL OF OCEANIC ENGINEERING, and IEEE/ACM TRANSACTIONS ON NETWORKING.

Muriel Médard (F'08) is a Professor in the Electrical Engineering and Computer Science at MIT. She was previously an Assistant Professor in the Electrical and Computer Engineering Department and a member of the Coordinated Science Laboratory at the University of Illinois Urbana-Champaign. From 1995 to 1998, she was a Staff Member at MIT Lincoln Laboratory in the Optical Communications and the Advanced Networking Groups. Professor Médard received B.S. degrees in EECS and in Mathematics in 1989, a B.S. degree in Humanities in 1990, a M.S. degree in EE 1991, and a Sc D. degree in EE in 1995, all from the Massachusetts Institute of Technology (MIT), Cambridge. She is an associate editor for the IEEE JOURNAL OF LIGHTWAVE TECHNOLOGY. She has served as an Associate Editor for the Optical Communications and Networking Series of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, as an Associate Editor in Communications for the IEEE TRANSACTIONS ON INFORMATION THEORY and as an Associate Editor for the *OSA Journal of Optical Networking*. She has served as a Guest Editor for the IEEE JOURNAL OF LIGHTWAVE TECHNOLOGY, the Joint special issue of the IEEE TRANSACTIONS ON INFORMATION THEORY and the IEEE/ACM TRANSACTIONS ON NETWORKING ON NETWORKING AND INFORMATION THEORY and the IEEE TRANSACTIONS ON INFORMATION FORENSIC AND SECURITY: Special Issue on Statistical Methods for Network Security and Forensics. She is a member of the Board of Governors of the IEEE Information Theory Society. She has served as TPC co-chair of ISIT, WiOpt and CONEXT. Professor Médard's research interests are in the areas of network coding and reliable communications, particularly for optical and wireless networks. She was awarded the 2009 Communication Society and Information Theory Society Joint Paper Award for the paper: Tracey Ho, Muriel Médard, Ralf Koetter, David Karger, Michelle Effros Jun Shi, Ben Leong, "A Random Linear Network Coding Approach to Multicast," IEEE TRANSACTIONS ON INFORMATION THEORY, vol. 52, no. 10, pp. 4413–4430, October 2006. She was awarded the 2009 William R. Bennett Prize in the Field of Communications Networking for the paper: Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Médard, Jon Crowcroft, "XORs in the Air: Practical Wireless Network Coding", IEEE/ACM TRANSACTIONS ON NETWORKING, Volume 16, Issue 3, June 2008, pp. 497–510. She was awarded the IEEE Leon K. Kirchmayer Prize Paper Award 2002 for her paper, "The Effect Upon Channel Capacity in Wireless Communications of Perfect and Imperfect Knowledge of the Channel," IEEE TRANSACTIONS ON INFORMATION THEORY, Volume 46 Issue 3, May 2000, Pages: 935–946. She was co- awarded the Best Paper Award for G. Weichenberg, V. Chan, M. Médard,"Reliable Architectures for Networks Under Stress," Fourth International Workshop on the Design of Reliable Communication Networks (DRCN 2003), October 2003, Banff, Alberta, Canada. She received a NSF Career Award in 2001 and was co-winner 2004 Harold E. Edgerton Faculty Achievement Award, established in 1982 to honor junior faculty members "for distinction in research, teaching and service to the MIT community." She was named a 2007 Gilbreth Lecturer by the National Academy of Engineering. Professor Médard is a Fellow of IEEE.

Milica Stojanovic (SM'08–F'10) graduated from the University of Belgrade, Serbia, in 1988, and received the M.S. and Ph.D. degrees in electrical engineering from Northeastern University, Boston, MA, in 1991 and 1993. After a number of years with the Massachusetts Institute of Technology, where she was a Principal Scientist, she joined the faculty of Electrical and Computer Engineering Department at Northeastern University in 2008. She is also a Guest Investigator at the Woods Hole Oceanographic Institution, and a Visiting Scientist at MIT. Her research interests include digital communications theory, statistical signal processing and wireless networks, and their applications to underwater acoustic communication systems. Milica is an Associate Editor for the IEEE JOURNAL OF OCEANIC ENGINEERING and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.